Investigation of Speech Coding Algorithms for Voice Communication through Personal Computers

by

Shahid Parvez

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING

May, 1998

INVESTIGATION OF SPEECH CODING ALGORITHMS FOR VOICE COMMUNICATION THROUGH PERSONAL COMPUTERS

BY

SHAHID PARVEZ

A Thesis Presented to the

FACULTY OF THE COLLEGE OF GRADUATE STUDIES

KING FAHD UNIVERSITY OF PETROLEUM & MINERALS

DHAHRAN, SAUDI ARABIA

In Partial Fulfillment of the
Requirements for the Degree of

MASTER OF SCIENCE

In

SYSTEMS ENGINEERING

MAY 1998

UMI Number: 1391786

KING FAHD UNIVERSITY OF PETROLEUM AND MINERALS

DHAHRAN 31261, SAUDI ARABIA

COLLEGE OF GRADUATE STUDIES

This thesis, written by

SHAHID PARVEZ

under the direction of his Thesis Advisor and approved by his Thesis Committee, has been presented to and accepted by the Dean of the College of Graduate Studies, in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN SYSTEMS ENGINEERING

Thesis Committee

M. Elshafei

Dr. M. A. Elshafei (Chairman)

Md. Shahgir Ahmed

Dr. M. S. Ahmed (Member)

June 1999

Dr. H. AlMohammed (Member)

Dr. O. Toker (Member)

Department Chairman

Dean, College of Graduate Studies

15/6/99

Date

This work is dedicated

to

my Parents,

Brother and Sister.

Acknowledgements

All praise be to Allah Subhanahu Wa Taala for his blessings and guidance.

My acknowledgements are due to the University of Petroleum & Minerals for support of this research work.

I wish to express my sincere appreciation to Dr. Moustafa Elshafei Ahmed, my major thesis advisor, for his continuous direction, help and encouragement. He was available to help me at all times.

I would also like to express my appreciation to my thesis committee comprising of Dr. Shahgir Ahmed, Dr. H. AlMohammad and Dr. Oner Toker for their support and encouragement.

I wish to thank my mother, brother and sister for all their moral support and sacrifices.

Finally, I would like to thank all my colleagues and members of the North Compound who have made my stay a memorable one.

Contents

List of Tables

List of Figures

Nomenclature

English Symbols

s	Speech signal
A	LPC analysis filter
M	Order of LPC filter
a	LPC coefficients
x	Line Spectrum pair
y	Filtered adaptive codebook vector
t	Target signal
h	Impulse response of the weighted synthesis filter
c	Fixed codebook vector
v	Adaptive codebook vector
u	Excitation signal
g	Gain
r	Autocorrelation coefficients

R	Autocorrelation Matrix

Greek Symbols

ϵ	Fixed perturbation vector
ω	Line spectral frequency
γ	Correction factor

Abbreviations

WSS	Wide Sense Stationary
LP	Linear Prediction
LPC	Linear Predictive Coding
CELP	Code Excited Linear Prediction
LSF	Line Spectral Frequencies
LSP	Line Spectrum Pairs
LBG	Linde Buzo and Gray
CS-ACELP	Conjugate Structure - Algebraic Code Excited Linear Prediction
CCITT	International Telephone and Telegraph Consultative Committee
ITU-T	International Telecommunications Union

Subscripts

n	Order of the polynomial

Superscript

$'$	Transpose

THESIS ABSTRACT

Name: SHAHID PARVEZ

Title: INVESTIGATION OF SPEECH CODING ALGORITHMS FOR VOICE COMMUNICATION THROUGH PERSONAL COMPUTERS

Major Field: SYSTEMS ENGINEERING

Date of Degree: MAY 1998

The tremendous growth of multimedia communications has led to an increasing demand for speech coding techniques. Low-cost highly-efficient speech coding at low bit rate is important for personal multimedia communications. Recent developments in speech coding around 8kb/s are investigated; in particular the work was focused on adapting the new standard G.729 for multimedia applications. A new speech coding algorithm compatible with PC environment for multimedia applications has been developed by adapting G.729 and enhancing it with the latest developments in mediumband coding. This coder at a sampling rate of 11ks/s and a bit rate of 8.8kb/s. provides high quality voice over the band of frequencies ranging from 20Hz to 5400Hz. It is computationally more efficient. This algorithm is compared to the standard G.729 with respect to spectral distortion and quality. Issues related to generation of codebooks and quantization of parameters are considered. An algorithm for efficient computation of the Line Spectral Frequencies is also proposed. The applications of the proposed coder include speech storage and communication in PC multimedia environment.

Keywords: Speech coding. Vector quantization, LSF, LPC, Multimedia.

Master of Science Degree

King Fahd University of Petroleum and Minerals
Dhahran, Saudi Arabia.

May, 1998.

خلاصة الرسالة

اسم الطـالب : شاهيد برفيز

عنوان الدراسـة : دراسة بحثية لخوارزميات ترميز المحادثات للأتصالات الصوتية من خلال الحواسب الشخصية.

التخـصص : هندسة النظم

تاريخ الشـهادة : صفر ١٤١٩ هـ

أدى التطور الهائل فى اتصالات الوسائط المتعددة إلى ازدياد الطلب على طرق جديدة لترميز الصوت. و لذلك فإن ترميز الصوت بكفاءة عالية و بتكلفة قليلة له أهمية كبيرة فى مجال الأتصالات الشخصية المتعددة الوسائط. وقد تم فى هذا البحث استعراض التطورات الحديثة فى نظم ترميز عند ٨ آلاف عينة/ثانية. و قد ركز البحث بصفة خاصة على تطوير النظام القياسى ج ٧٢٩ للعمل فى تطبيقات الأتصالات المتعددة الوسائط.

وقد أدى هذا البحث إلى استنباط خوارزمى جديد لترميز الأصوات مبنى على ج ٧٢٩ مع تحسينه بإدخال بعض التطورات التى ظهرت حديثا فى مجال ترميز المحادثات عند الطيف المتوسط. ويعمل نظام الترميز المقترح على ١١ إلف عينة/ثانية وبمعدل ٨،٨ الاف نبرة/ثانية ويعطى جودة عالية فى منطقة طيف ممتدة من ٢٠ زبزبة/ثانية إلى ٥٤٠٠ زبزبة/ثانية. كما حقق كفاءة حسابية أعلى من ج ٧٢٩. وكذلك تم مقارنة نظام الترميز المقترح بالنظام القياسى ج ٧٢٩ من حيث التشويه الطيفى و جودة الصوت. و عالج البحث ايضا الموضوعات المتعلقة بجداول الترميز و تكميم المعاملات. كما تم أقتراح برمجيات لحساب زبزبات الطيف الخطية بكفاءة حسابية عالية.

و تشمل تطبيقات نظام الترميز المقترح تخزين الأصوات و الأتصالات بالوسائط المتعددة من خلال الحاسب الشخصى.

درجة الماجستير فى العلوم

جامعة الملك فهد للبترول والمعادن

الظهران المملكة العربية السعودية.

صفر ١٤١٩ هـ

Chapter 1

INTRODUCTION

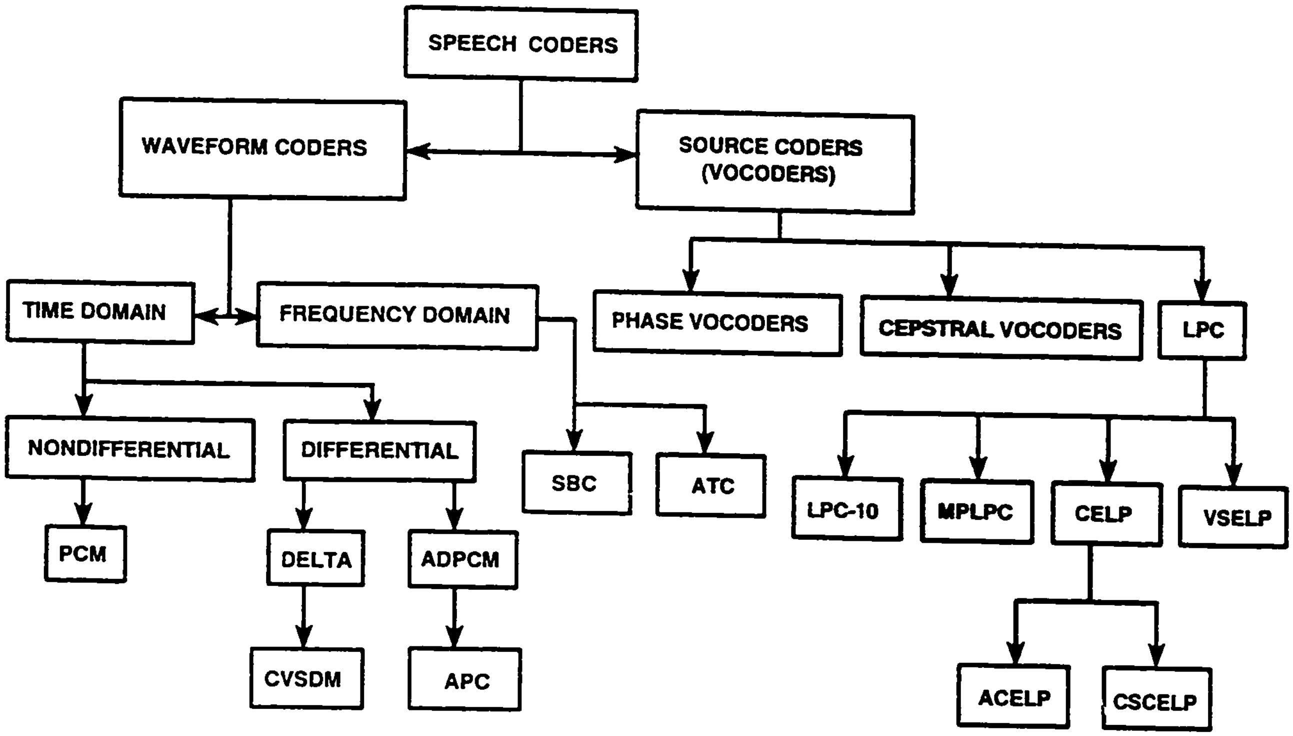
1.1 Introduction

This chapter gives an introduction to the main features of the thesis work. In section 1.2 the readers are introduced to speech coding and the basic techniques being applied in speech coding. Speech coding cannot be appreciated without some basic understanding of the signal i.e.. speech involved. This is the focus of section 1.3 which introduces basic speech signals and the speech production model. In section 1.4 motivation for this thesis work is given and finally in section 1.5 an outline of the work is presented.

1.2 Introduction to Speech Coding

An integral part of multimedia communications is the speech coder. Speech coding refers to the process of reducing the bit rate of digital speech representations for transmission or storage, while maintaining a speech quality that is acceptable for application. Multimedia refers to having a variety of media presented either simultaneously or sequentially. Thus, speech coding for multimedia automatically implies that speech-coding bit-stream will be sharing the communication channel with other signals. Speech quality as produced by a speech coder is a function of bit rate, complexity, delay, and bandwidth. Hence, when considering speech coders it is important to review all these attributes. It is important to realize that there is a strong interaction between all these attributes and that they can be traded off against each other. For example, lower-bit-rate coders tend to have more delay than higher-bit-rate coders. They may also require higher complexity to implement and often have lower quality than the higher-bit-rate coders.

There are basically two class of speech coders, namely, waveform and source coders [1]. The hierarchy of speech coders is shown in Fig. 1.1. Speech coders differ widely in their approaches to achieving signal compression. Based on the means by which they achieve compression, speech coders are broadly classified into two categories: *Waveform coders* and *Source coders* [2]. Waveform coders essentially strive to reproduce the time waveform of the speech signal as closely as possible. They are



NOTATION

PCM	:	PULSE CODE MODULATION
CVSDM	:	CONTINUOUSLY VARIABLE SLOPE DELTA MODULATION
APC	:	ADAPTIVE PREDICTIVE CODING
SBC	:	SUB-BAND CODING
ATC	:	ADAPTIVE TRANSFORM CODING
LPC	:	LINEAR PREDICTIVE CODER
VOCODERS	:	VOICE CODERS
MPLPC	:	MULTI-PULSE LINEAR PREDICTIVE CODING
CELP	:	CODE EXCITED LINEAR PREDICTION
ACELP	:	ALGEBRAIC CELP
CSCELP	:	CONJUGATE STRUCTURE CELP
VSELP	:	VECTOR SUM EXCITED LINEAR PREDICTION

Figure 1.1: Hierarchy of speech coders

in principle, designed to be source independent and hence code equally well a variety of signals. They have the advantage of being robust for a wide range of speech characteristics and for noisy environments. Examples of waveform coders include *Pulse Code Modulation* (PCM), *Differential Pulse Code Modulation* (DPCM), *Adaptive Differential Pulse Code Modulation* (ADPCM), *Delta Modulation* (DM), *Continuously Variable Slope Delta Modulation* (CVSDM), and *Adaptive Predictive Coding* (APC) [3]. Source coders on the other hand achieve very high economy in transmission bit rate, and are in general more complex. While waveform coding is more suitable for high bit rate applications, source coding is suitable for low to medium bit rate applications. Some of the source coders are the *Phase Vocoders*, *Cepstral Vocoders* and the *Linear Predictive Coders*. There are many coders based on Linear Predictive Coding technique namely the *LPC-10*, *Multipulse Linear Predictive Coders* (MPLPC), *Vector Sum Excited Linear Prediction Coders* (VSELP) and the *Code Excited Linear Predictive Coders* (CELP). The CELP technique has so far been the most proliferent technique for low-bit-rate coding.

Low-cost highly-efficient speech coding at low bit rate is important for personal multimedia communications [4]. In particular demands for real-time speech coding on personal computers and transmission of codes via modems are increasing [5]. Therefore the focus of this work is on speech coders which lie in the class of source coders. Since these are based on an apriori knowledge of the signal to be encoded we will provide a brief introduction to the speech signal.

1.3 An Introduction to Speech

Speech signals are basically non-linear and time variant in nature. However over a short period of time they can be approximated as wide sense stationary (WSS) signals. They are composed of a sequence of sounds. These sounds and the transition between them serve as a symbolic representation of information. The acoustical speech waveform is an acoustic pressure wave which originates from voluntary physiological movements of the glottis, vocal tract, velum, etc.

The two basic kinds of sounds are the voiced and the unvoiced sounds. Voiced sounds are produced by forcing air through the glottis with the tension of the vocal tract adjusted so that they vibrate in a relaxation oscillation, thereby producing quasi-periodic pulses of air which excite the vocal tract. Unvoiced or fricative sounds are generated by forcing air through a constriction at some point in the vocal tract at a high velocity to produce turbulence.

Since the vocal tract shape is varied as a function of time to produce the desired speech sounds for communication, so must the spectral properties of speech signal vary with time. In contemporary signal processing, it is possible to exhibit these changes using a three-dimensional plot of magnitude spectra over time. However, another technique to view time-varying spectral characteristics of speech is through the use of speech spectrogram. Spectrogram of a word sounding "matlab" for .5 sec, sampled at a rate of 8192Hz is shown in Fig. 1.3 Darkness of the output

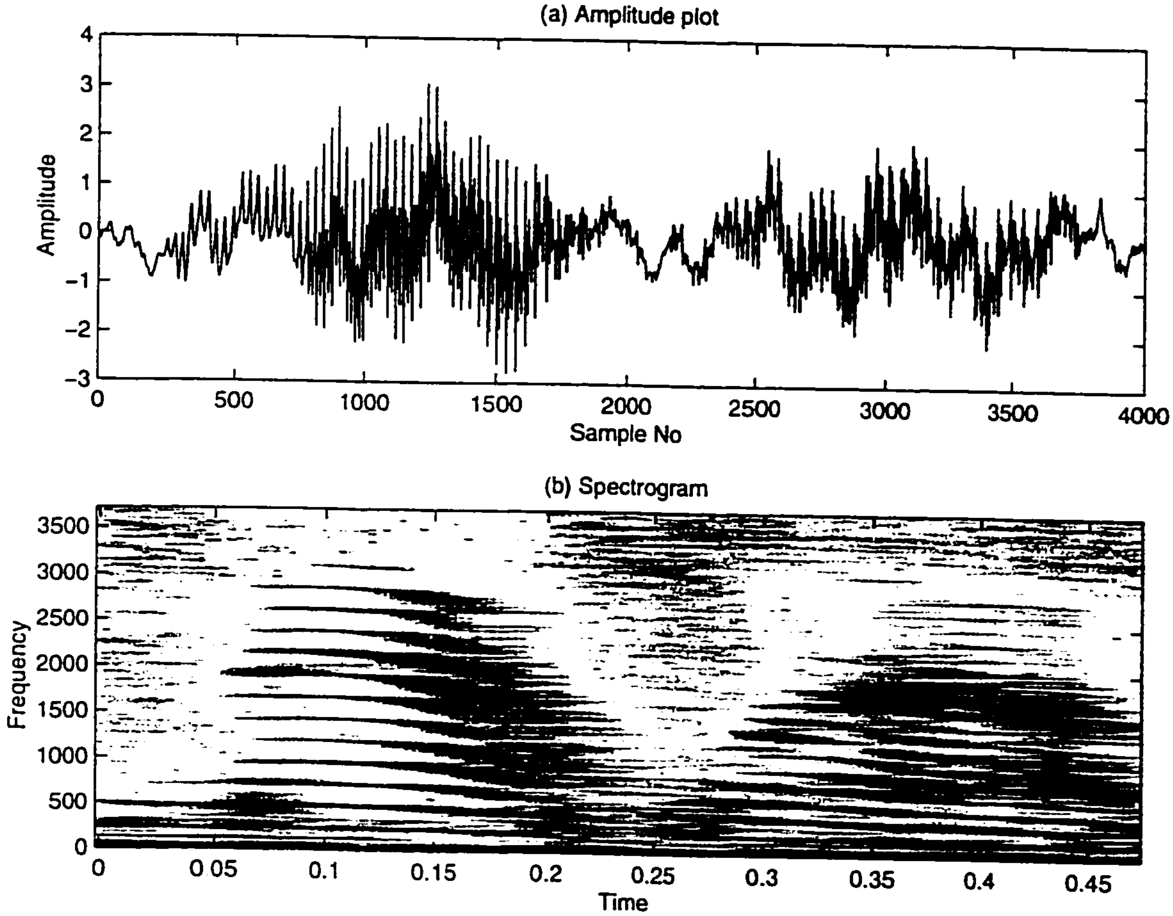


Figure 1.2: (a) Magnitude spectrum for the word "matlab" (b) Spectrogram for "matlab"

markings reflects the relative amplitude from (time vs. frequency) filter. Peaks in the spectrum such as formant frequencies appear as dark horizontal bands in the spectrogram. Voiced speech is characterized by equally spaced horizontal striations due to the periodic nature of the glottal excitation. Unvoiced speech sounds, due to their noiselike excitation, are characterized by rectangular dark patterns at low frequencies, with somewhat random occurrences of light spots due to sudden variations in energy. Spectrograms can only represent spectral amplitude, with no means for illustrating phase.

The linear prediction model can be related to the speech production model, with the significant feature that the parameters of the speech production model are easily obtained using linear mathematics. Further in case of autoregressive models, the linear prediction coefficients can be used in the efficient representation of segments of speech in terms of very few parameters, namely the coefficients of all pole infinite impulse response(IIR) filter.

1.3.1 Speech Production Model

A general discrete-time model for speech production is shown in Fig. 1.3.1. This model is called a terminal-analog model, meaning that the signals and systems involved in the model are only superficially analogous, if at all, to the true physical system except at the terminus, where both produce analogous waveform, speech. In other words this model attempts to represent the speech production process based on its output signal characteristics. In this system a vocal tract model and radiation model are excited by a discrete-time glottal excitation signal. During unvoiced speech activity, the excitation source is a flat spectrum noise source modeled by random noise generator. During periods of voiced speech activity, the excitation uses an estimate of the local pitch period to set an impulse train generator that drives a glottal pulse shaping filter[3]. This model was the basis of the early coders such as the LPC-10 and LPC-10e. In fact the basic philosophy of its structure continued to evolve to become a major class of the new generation of coders.

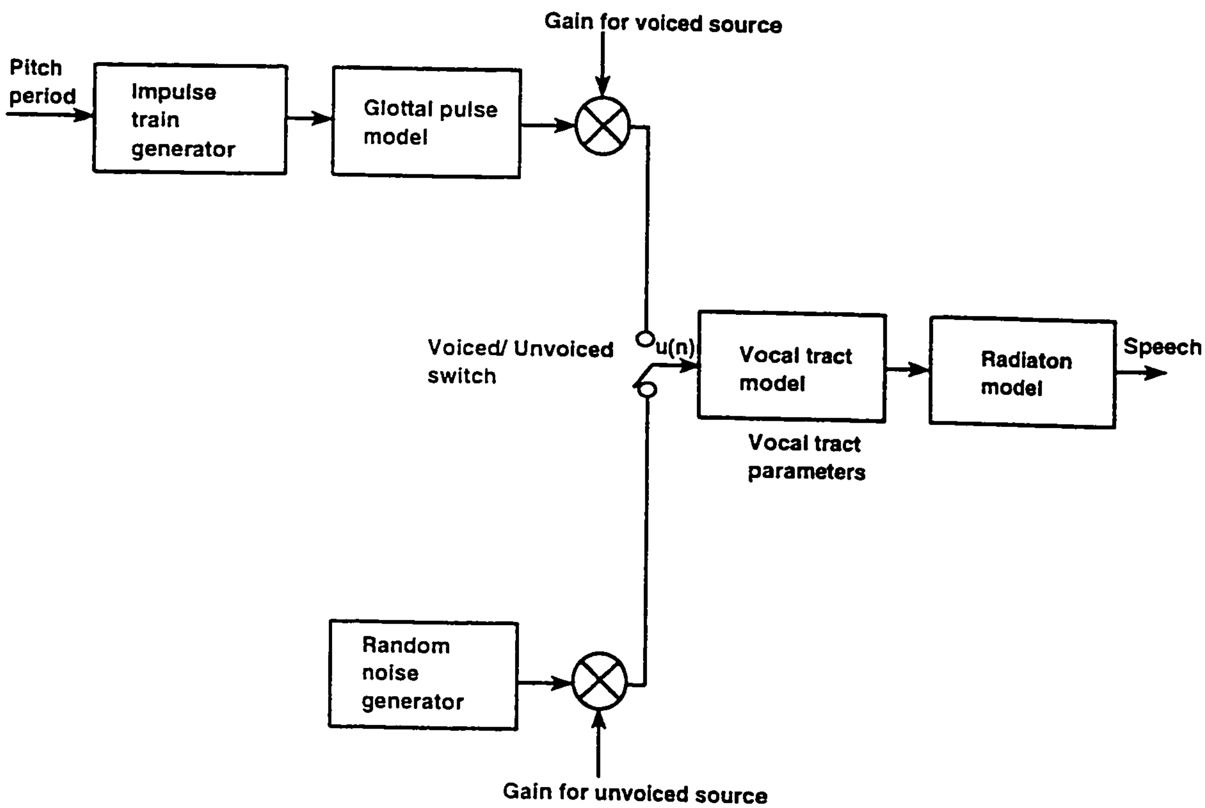


Figure 1.3: Speech Production Model

1.4 Motivation for this Work

There is a need for a speech coder at 11ks/s compatible with the Personal Computers and Microsoft platforms for multimedia applications. These applications include. communication via modems installed on PC. coding of compressed speech over CD-players etc. As the speech-coding bit-stream will be sharing the communication channel with other signals. we needed to code the speech at lowest possible bit rate without much deterioration in quality. The frequency range of our application ranges from 20 Hz to around 5400 Hz. hence the coder needed for our application is a source coder. Recent years have witnessed a breakthrough in the development of speech coding techniques. however. most of the research has focussed on narrow-band speech signals where the transmission bandwidth is limited to 300-3400 Hz. Although this bandwidth limitation is acceptable in telephone systems. it degrades the quality in the above mentioned applications where the speech is to be heard through high quality loudspeakers. For these systems. a bandwidth of 20-5400 Hz is more appropriate. and the sampling rate needed for this bandwidth is 11ks/s. which is compatible with the available platforms. Code Excited Linear Prediction (CELP) coding has proven to be the most promising algorithms for narrowband digital speech. However. few studies have attempted to apply CELP to the context of mediumband speech. The main drawback of CELP is its gross computational complexity. As the sampling frequency is doubled, larger frame sizes are needed to maintain a low bit rate transmission. Consequently, the use of much larger excitation codebooks becomes inevitable. For instance, if we assume the same proportional bit rates and block lengths, the typical codebook size increases from a thousand entries (10bits) to a million entries (20bits). Searching and storing such a codebook size is rather impractical, unless some suboptimal approach is utilized [6].

From the above discussion, it seems that it is impossible to use an exhaustive search approach for CELP coding of mediumband speech. However, the Algebraic Code Excited Linear Prediction (ACELP) approach of the recently standardized speech coder G.729, with its other important features, offers the solution to this problem. With this background, this work was focussed on developing a medium band speech coder by adapting the standard G.729 for the required applications.

1.5 Outline of Work

This chapter was dedicated to a brief introduction of speech and speech coding. The goal of our work was to adapt the standard G.729 to multimedia applications. The recently standardized ITU-T Recommendation G.729 [7], at 8kb/s was first implemented off-line. However exact implementation of the standard is not guaranteed due to the non availability of complete information on the standard. Further, the codebooks are designed based on the data available at our disposal. It has been found that the vector quantization of the LSF parameters, fixed-codebook search and gain quantization involves lot of complex search procedures thereby increasing the computational time. An effort was made to reduce this computational time with a reasonable tradeoff with speech quality. Some basic fundamentals like the LPC analysis and the CELP technique are introduced in chapter 2, along with a brief summary of the literature review. An overview of the standard G.729 is given in the later part of chapter 2.

Our application for communication through computer requires sampling rate of 11ks/s to deal with ".wav" and other files. A new coder based on CELP is proposed with a sampling rate of 11ks/s and a bit rate of 8.8kb/s. Further details are the issue of the third chapter.

An efficient algorithm for computing the Line Spectral Frequencies was developed using previous results which exploit the symmetry of the LSF polynomials, as well as some results regarding the maximum level of precision necessary for maintaining good perceived quality in synthetic speech produced from LSP frequencies. This algorithm and two techniques based on this algorithm are presented in chapter 4. Computational details are outlined in Appendices A, B, and C.

Results and discussions are made in chapter 5. The proposed coder is compared with the standard G.729. Two other quantization schemes are tested with the proposed coder and is compared to the standard in terms of the spectral distortion measure. Efficiency of the proposed algorithm for LSF computation is checked. Conclusions and recommendations are made in chapter 6.

Chapter 2

BACKGROUND AND LITERATURE REVIEW

2.1 Introduction

Until recently, low-rate algorithms were of interest only to researchers in the field. Speech coding is now of interest to many engineers who are confronted with the difficult task of learning the essentials of voice compression in order to solve implementation problems, such as fitting an algorithm to an existing fixed point signal processor or developing low-power single-chip solutions for portable cellular telephones.

This chapter is devoted to giving a brief background of important features of speech coding. Simultaneously literature review of the existing speech coding techniques is done in section 2.2. In section 2.3 the standard speech coder G.729 is briefly discussed. With the available information and requirements on hand, the possible main features of the required coder are summarized in section 2.4.

2.2 Background and Review of Existing Techniques

In this section we deal with some important aspects involved in a source coder. At the same time we will highlight the contributions made within their prospective.

2.2.1 Linear Prediction Analysis

The source system model became associated with the Autoregressive (AR) time-series methods where the vocal tract filter is all-pole and its parameters are obtained by linear prediction analysis [8]: a process where the present speech sample is predicted by the linear combination of previous samples. Itakura and Saito [9], and Atal and Schroeder [10], were the first to apply Linear Prediction (LP) techniques to speech. Atal and Hanauer [11] later reported an analysis-synthesis system based on LP. Theoretical and practical aspects of LPC and the problem of spectral analysis of speech using linear prediction was addressed by Markel and Gray [12].

Let $s(n)$ be a WSS random process i.e. speech. Now suppose we want to predict the value of the sample $s(n)$ using a linear combination of M most recent past samples. The estimate is given by

$$\hat{s}_M(n) = -\sum_{i=1}^{M} a_i s(n-i). \tag{2.1}$$

the integer M is called the prediction order and the coefficients a_i are called the Linear Prediction Coefficients (LPC). The estimation error is

$$\begin{aligned} e_N(n) &= s(n) - \hat{s}_M(n), \\ &= s(n) + \sum_{i=1}^{M} a_i s(n-i). \end{aligned} \tag{2.2}$$

Taking the Z-transform of the above equation we get

$$E(z) = S(z)A(z), \tag{2.3}$$

where

$$A(z) = 1 + \sum_{i=1}^{M} a_i z^{-i}. \tag{2.4}$$

$A(z)$ is called the linear prediction filter. The spectral envelope of a speech of length L samples can be approximated by the transmission function of an all-pole digital filter given by $1/A(z)$.

2.2.2 Line Spectral Frequencies

The concept of the line spectral frequencies LSFs ($0 < \omega < \pi$) was first introduced by Itakura [9]. Line Spectral Frequencies (LSFs) are a transformation of the LPC parameters. The LSFs can be interpreted as the resonant frequencies of the vocal tract under the two extreme boundary conditions at the glottis (complete closure and complete opening). Some important properties of these LSFs make them an efficient coding tool. They are discussed in detail in the fourth chapter.

2.2.3 Related Work on Speech Coding

LP is not the only method used for source-system analysis. Homomorphic analysis, a method that can be used for separating signals that have been combined by convolution, has also been used for speech analysis. Oppenheim [13], was a strong proponent of this method. One of the inherent advantage of homomorphic speech analysis is the availability of pitch information from the cepstrum.

The emergence of VLSI technologies along with advances in the theory of digital signal processing during the 1960's and 1970's provided even more incentives for getting new and improved solutions to the speech coding problem. Analysis-synthesis of speech using the Short Time Fourier Transform (STFT) was proposed by Rabiner and Schafer [14]. In the mid-to late 1970's there was also continued activity in linear prediction and transform coding. During the 1970's, there was a parallel effort made for the application of linear prediction in military secure communications. A federal standard (FS-1015), which is based on the LPC-10 algorithm, was developed in the early 1980's.

Research efforts in the 1980's and 1990's have been focused upon developing robust low-rate speech coders capable of producing high-quality speech for communication applications. Most of this work was driven by the need for narrow-band and secure transmission in cellular and military communications. Competing methodologies were promoted using multipulse and vector excitation schemes in LPC by Atal et al. [15], employing vector quantization (VQ) promoted by Gray [16]. Vector quantization proved to be very useful in encoding LPC parameters [17], [18], [19]. Later on LPC to LSF transformation was introduced by Itakura [9] and its features investigated by others [20], [21]. Vector quantization of the LSF parameters further increased the rate of compression by reducing the number of bits required for transmission. Vector quantization of the LSF parameters itself has been a point of great research till date and efficient coding schemes have been promoted as a result [22].

2.2.4 Analysis by Synthesis Search Procedure and CELP Model

Atal and Schroeder [23], proposed a linear prediction algorithm with stochastic vector excitation which they called "Code Excited Linear Prediction"(CELP). Speech is generated by exciting the short term filter or LPC filter and the long term filter or pitch filter by proper excitation signal. In the analysis by synthesis technique the speech is reconstructed at the encoder itself and the excitation signal is determined by minimizing the perceptually weighted error between the original and the synthesized speech. CELP coders are also called hybrid coders because they combine the features of traditional vocoders with the wave-form matching features of the waveform coders. Although the first paper on CELP addressed the feasibility of vector excitation coding, follow-up work demonstrated that CELP coders were capable of producing medium-rate and low-rate speech adequate for communication applications. Real-time implementation of hybrid coders became feasible with the development of highly structured codebooks [24].

We will now discuss the basic CELP model which is the basis for our coder. In CELP the excitation sequence is selected from a codebook of zero-mean Gaussian sequences. The block-diagram of the CELP coder is shown in Fig. 2.1. It consists of the cascade of two all-pole filters, with coefficients that are updated periodically. The first filter is a long-delay pitch filter used to generate the pitch periodicity in voiced speech. Its parameters can be determined by minimizing the prediction error energy, after pitch estimation, over a frame duration of 5 msec. The second filter is a short-delay all-pole (vocal-tract) filter used to generate the spectral envelope (formants) of the speech signal. This filter usually has 10-12 coefficients that are determined periodically using the LP analysis as discussed earlier.

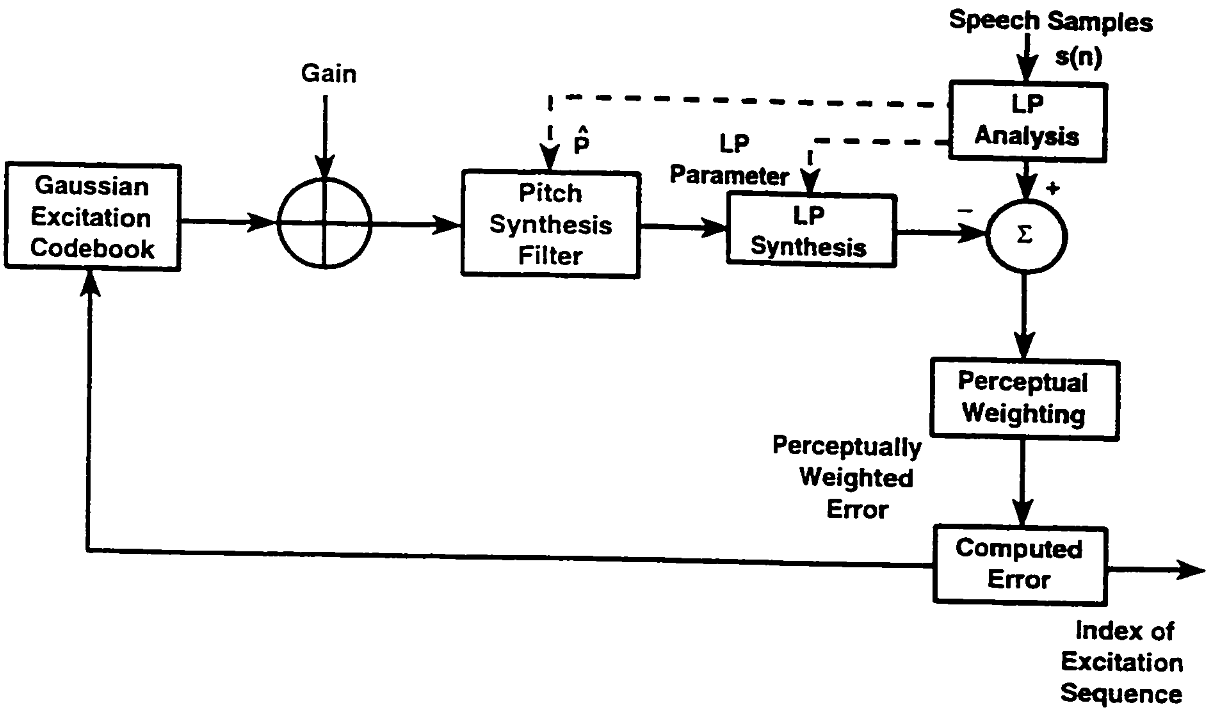


Figure 2.1: CELP analysis-by-synthesis coder

A stored sequence from a Gaussian excitation codebook is scaled and used to excite the cascade of a pitch synthesis filter and the LPC synthesis filter (computed over the current frame). The synthesized speech is compared with the original speech and the difference constitutes the residual error signal, which is perceptually weighted by passing it through a filter that is characterized by the system function. This perceptually weighted error is squared and summed over a subframe block to give the error energy. By performing an exhaustive search through the codebook we find the excitation sequence that minimized the error energy. The gain factor for scaling the excitation sequence is determined for each codeword in the codebook by minimizing the error energy for the block of samples.

2.2.5 Recent Standardization Activities

Progress in speech coding, particularly in the late 1980's enabled recent adoptions of low-rate algorithms for mobile telephony. An 8kbit/s hybrid coder has been selected for the North American digital cellular standard, and a similar algorithm has been selected for the 6.7-kbit/s Japanese cellular standard. In Europe, a standard that used a 13-kbit/s regular pulse excitation algorithm has been deployed by the "Group Speciale Mobile" (GSM). Parallel standardization efforts for secure military communications have resulted in the adoption of 4.8kb/s hybrid algorithm for federal Standard 1016 [25]. Other methods have also been standardized by the International Telephone and Telegraph Consultative Committee (CCITT) within study group. XV. recently renamed the International Telecommunications Union - Telecommunications standardization sector (ITU-T) study group (SG)15 . These are Recommendations G.711 on 64kb/s PCM, Recommendations G.726 on 40, 32, 24, and 16kb/s ADPCM, Recommendation G.728 on 16kb/s LD-CELP and more recently Recommendation G.729 CS-ACELP [7].

The Moving Pictures Expert Group (MPEG) within the International Organization of Standards (ISO) has developed a series of audio-visual standards known as MPEG-1(standardized in November 1992) and MPEG-2 (standardized in November 1994) [26]. These audio coding standards are the first international standards in the field of high-quality digital audio compression. MPEG-1 covers coding of stereophonic audio signals at high sampling rates aiming at transparent quality, whereas MPEG-2 offers stereophonic audio coding at lower sampling rates. The MPEG-I operates in single-channel or two-channel stereo modes at sampling frequencies of 32KHz, 44.1KHz, and 42KHz. MPEG-2 provides audio coding at sampling frequencies 16KHz, 22.5KHz, and 24KHz. Speech and audio coding are similar in that, in both cases, quality is based on the properties of human auditory perception. On the other hand, speech can be coded very efficiently because a speech production model is available, whereas nothing similar exists for audio signals.

The International Telecommunications Union(ITU) has recently standardized three speech coders which are applicable to low-bit-rate multimedia communications. ITU Recommendation G.729, 8kb/s CS-ACELP has a 15 ms algorithmic coder delay and provides network-quality speech. It was originally designed for wireless applications, but is applicable to multimedia communications as well. Annex A of Rec. G.729 is a reduced-complexity version of the CS-ACELP coder. It was designed explicitly for simultaneous voice and data applications that are prevalent in low-bit-rate multimedia communications. These two coders can use the same bit-stream format and can interoperate. The ITU Rec. G.723.1 at 6.3 kb/s and 5.3 kb/s speech coder for multimedia communications was designed originally for low-bit-rate videophones [27]. Its frame size of 30ms and one-way algorithmic coder delay of 37.5 ms allow for a further reduction in bit rate compared to the G.729 coder. In applications where low delay is important, the delay of G.723.1 may be too large. However, if the delay is acceptable, G.723.1 provides a lower-complexity alternative to G.729 at the expense of a slight degradation in quality. Though these recommendations have been accepted as speech coding standards they are yet to be implemented for multimedia applications. This motivated us to work in this area, and study the issues related to implementation for personal computer applications. We will now give some detailed analysis of the standard G.729, which is later adapted for our applications.

2.3 Important Aspects of Standard G.729

This coder also called the Conjugate Structure - Algebraic Code Excited Linear Prediction (CS-ACELP) coder is based on six main features :

- Adaptive spectral shaping.
- LSF quantization using interframe prediction.
- Algebraic codebook excitation.
- VQ gain quantization with backward prediction.
- Post filtering and
- Frame erasure concealment.

This CS-ACELP coder is based on the code excited linear prediction coding model. The coder operates on 10-ms speech frames, each corresponding to 80 samples. In addition there is a look-ahead of 5 ms, resulting in a total arithmetic delay of 15 msec.

The encoding principle is shown in Fig. 2.2 Linear prediction (LP) analysis is done once per 10 ms frame to compute the LP filter coefficients. These coefficients are converted to Line Spectral Frequencies (LSFs) and quantized using predictive split two-stage vector quantization (VQ) with 18 bits. The short-term synthesis filter is based on a 10-th order LP filter. The excitation signal is chosen by using

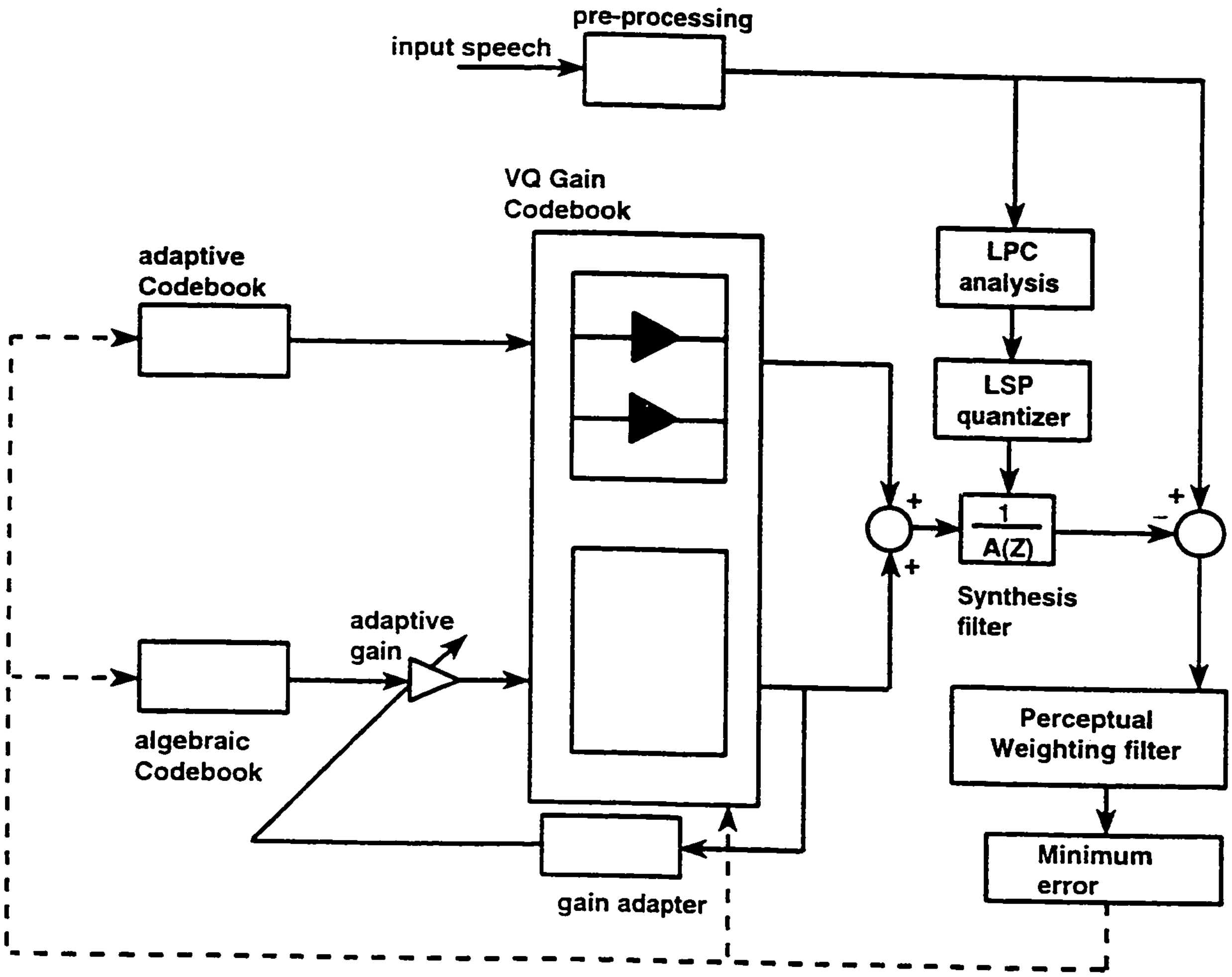


Figure 2.2: Encoder of G.729

an analysis-by-synthesis search procedure in which the error between the original and the reconstructed speech is minimized according to a perceptually weighted distortion measure.

The speech frame is divided into two 5-ms subframes(40 samples) each. The excitation parameters are determined for each subframe. The quantized and unquantized LP coefficients are used for the second subframe, whereas for the first subframe interpolated LP filter coefficients are used. The open-loop pitch is estimated once per 10-ms frame based on the perceptually weighted speech signal. Closed loop pitch delay with 1/3 resolution is used. The pitch delay is encoded with 8 bits in the first subframe and differentially encoded using 5 bits in the second subframe. An algebraic codebook with 17 bits is used for the fixed-codebook excitation. The gains of the adaptive and fixed-codebook are vector quantized with 7 bits using a conjugate structure codebook with MA prediction applied to the fixed-codebook gain.

The decoding principle is shown in Fig. 2.3. First, the parameter indices are extracted from the received bit stream. The speech is reconstructed by filtering the excitation through the LP synthesis filter. The reconstructed speech signal is passed through a post-processing stage, which includes an adaptive post filter based on the long-term and short-term synthesis filters, followed by a high-pass filter and scaling operation.

In the following sections, important features of the standard G. 729 are discussed.

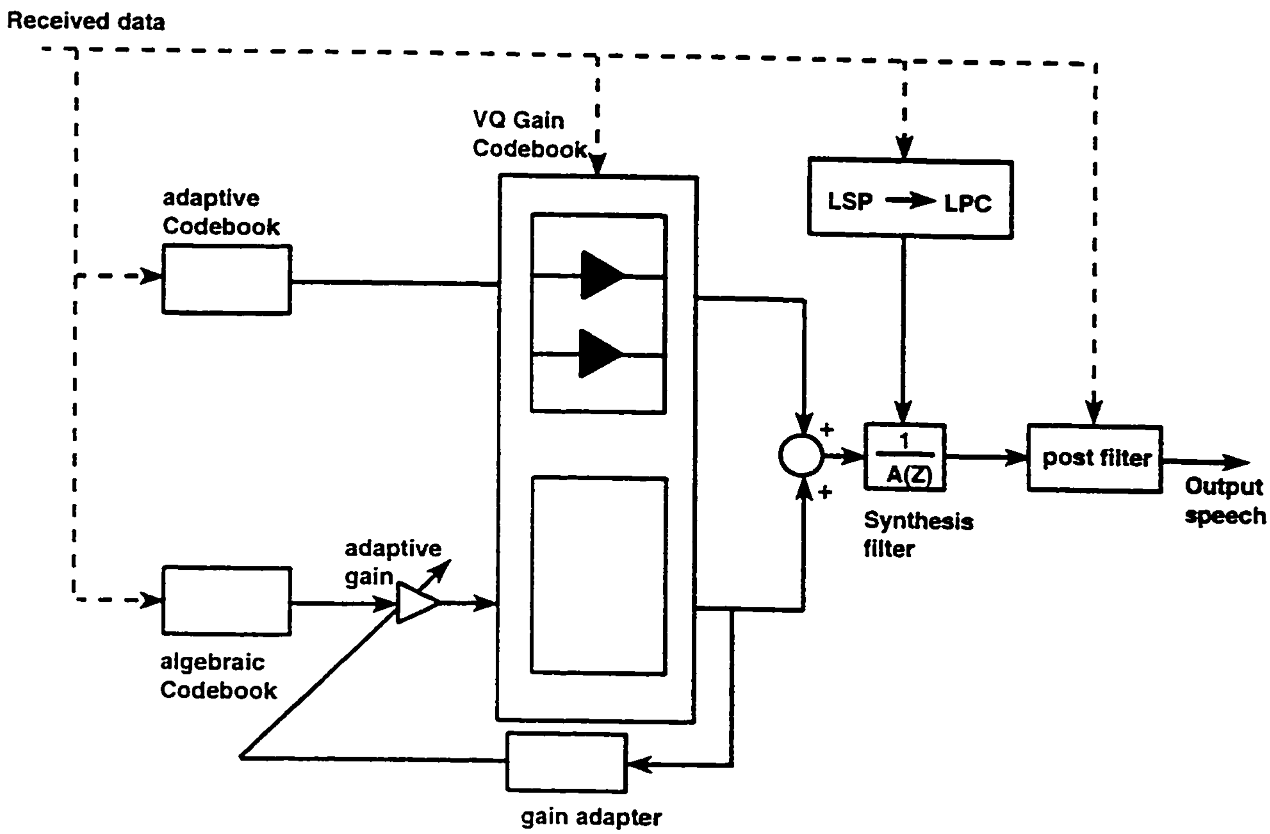


Figure 2.3: Decoder of G.729

2.3.1 Adaptive Spectral Shaping

The perceptual weighing filter is based on the unquantized LP filter coefficients a_i and is given by

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)}, \tag{2.5}$$

where A(z) is the LP analysis filter. The value of γ_1 and γ_2 determine the level of noise masking. These values are adapted using a criterion based on the minimum distance between two successive LSF coefficients [28].

2.3.2 LSF Quantization

The LSF quantization technique used in G. 729 has been discussed in detail as this technique is compared with the quantization technique of the proposed coder at a later stage.

In this technique the LSF parameters are quantized by using an interframe moving average prediction [29]. The nth frame quantized LSF parameters Ω_n are generated using

$$\Omega_n = G_0 C_n + \sum_{i=1}^{4} G_i C_{n-i} \; ; \quad \sum_{i=0}^{4} G_i = I \tag{2.6}$$

where C_n is the output vector from the two-stage VQ at the nth frame, I is the unit matrix , and G_i is a diagonal prediction matrix. G_i is given by

$$G_i = \begin{bmatrix} g_{1i} & \cdots & \cdots & 0 \\ \vdots & g_{2i} & & \vdots \\ \vdots & . & . & \vdots \\ \vdots & . & . & \vdots \\ 0 & \cdots & \cdots & g_{pi} \end{bmatrix} \tag{2.7}$$

where g_{pi} are the weighting coefficients. Two sets of Moving Average (MA) prediction matrices are trained during vector quantizer design. The predicted LSF is quantized using a two stage VQ with a split structure at the second stage. The resultant codevector at the nth frame is obtained as

$$C_n = C_{1j} + \begin{cases} C_{2j}^{L} & j < 5 \\ C_{2j}^{H} & j \geq 5 \end{cases} \tag{2.8}$$

Here, C_{1j} are the decoded values of the first stage, and C_{2j}^{L} and C_{2j}^{H} are the decoded values of the second stage. The first stage is a 10-th order vector, and the second stage consists of a high-order vector C_{2j}^{H} and a low order vector C_{2j}^{L}, where both these vectors are of fifth order. The first stage uses 7 bits, each second stage uses 5 bits, and the selection of predictive coefficients uses 1 bit.

2.3.3 Searching the LSF

The procedure for encoding the LSF parameters is as follows. For each set of the two MA predictors the following steps have to be accomplished.

1. The first codebook is searched and the index that minimizes the unweighted error between input LSF and the codebook vector is selected to give C_{1j} $1 \leq j \leq 10$.

2. This is followed by a search of the second codebook, first the lower part of the codebook is searched followed by the upper part by minimizing the weighted mean squared error. The codevector is generated using Eq. 2.8 and the quantized LSF is computed using Eq. 2.6. The weighted mean-squared error that is to be minimized is given by

$$d_{lsf} = \sum_{i=1}^{10} w_i(\omega_i - \Omega_{i,n})^2 \tag{2.9}$$

where ω_i is the ith element of the input LSF and $\Omega_{i,n}$ is the ith element of the quantized LSF at the nth frame given by Eq. 2.6. The weights are made adaptive as a function of the unquantized LSF coefficients ,

$$w_i = \begin{cases} 1 \text{ if } \omega_{i+1} - \omega_{i-1} - 1 > 0 \\ 10(\omega_{i+1} - \omega_{i-1} - 1)^2 + 1, \text{ otherwise} \end{cases} \tag{2.10}$$

where ω_i are the unquantized LSF parameters $\omega_0 = .04\pi$ and $\omega_{10} = .92\pi$.

The code indices and the MA coefficients index that minimizes the weighted error is selected to give the quantized LSF by Eq. 2.6.

2.3.4 Algebraic Codebook

A 17 bit algebraic codebook is used for the fixed excitation codebook. The excitation vector contains four non-zero pulses. These four pulses can take amplitudes and positions as shown in Table 2.1. The pulse positions of the first three pulses are each encoded with 3 bits. the fourth pulse is encoded with 4 bits, while the sign is encoded using 1 bit for each pulse. This gives a total of 17 bits for the four pulses. In this structure. 4 sign bits are determined through straightforward quantization whereas 13 position-bits are determined in an analysis-by-synthesis manner [30].

Pulse	Sign	Positions
i_0	s_0: ± 1	m_0 : 0,5,10,15,20,25,30,35
i_1	s_1: ± 1	m_1: 1,6,11,16,21,26,31,36
i_2	s_2: ± 1	m_2: 2,7,12,17,22,27,32,37
i_3	s_3: ± 1	m_3: 3,8,13,18,23,28,33,38 4,9,14,19,24,29,34,39

Table 2.1: Structure of the fixed codebook for G.729

2.3.5 Quantization of Gain

The gains of the adaptive excitation and the fixed excitation vectors are determined by using the 7-bit VQ-conjugate structure gain codebook. Although the pitch(adaptive) gain is generally distributed around 1, the fixed-codebook gain has a wider range. To compensate for this wider range 4th-order MA gain predictor with fixed coefficients is used [29].

Vector quantization of the gains is done using a 2-stage conjugate structure codebook [31, 32, 33]. The first stage consists of a 3 bit two dimensional codebook and the second stage consists of a 4 bit two dimensional codebook. While the first entry in each codebook corresponds to pitch gain, the second entry corresponds to a correction factor resulting from MA prediction of fixed codebook gain. The appropriate gain codebook index is selected by minimizing distortion [29].

2.3.6 Postfiltering

The postfilter used in the standard G.729 is the cascade of three filters: a long-term postfilter, a short-term postfilter, and a tilt compensation filter [28].

2.3.7 Error Concealment

Error concealment is accomplished by passing a parity bit from the encoder. Before the excitation is reconstructed at the decoder, the parity bit is recomputed. If this bit is not identical to the transmitted parity bit, it is likely that bit error occurred during transmission. When this happens the concealment procedure of CS-ACELP reconstructs the current frame by using previously received information.

2.4 Description of Proposed Coder

Having seen the basic functionalities of speech coder like the LPC analysis, analysis by synthesis technique, and also the standard G.729 coder we are in a position to suggest a suitable configuration for the coder required for multimedia applications. The frequency range of our application ranges from 50 Hz to around 5000 Hz, hence the coder needed for our application is a source coder. The coder proposed in this work is based on Code Excited Linear Prediction (CELP)[23]. CELP coding is a frame-oriented technique that breaks a sampled input signal into blocks of samples (i.e., vectors) that are processed as one unit. CELP coding is based on analysis-by-synthesis search procedures, perceptually weighted vector quantization(VQ), and linear prediction(LP). An LP filter is used to model the speech signal's short-term spectrum, or formant structure. Long-term signal periodicity, or pitch is modeled by an adaptive codebook. The residual from the short-term LP and pitch is vector quantized using a fixed stochastic code book. The optimal scaled excitation vectors from adaptive and stochastic codebooks are selected by minimizing a time varying, perceptually weighted distortion measure that improves subjective speech quality by exploiting masking properties of human hearing.

The basic features of this coder are

- Line Spectral Frequency (LSF) vector quantization
- Algebraic codebook excitation
- Gain vector quantization

Further details about the coder are discussed in the next chapter.

Chapter 3

PROPOSED SPEECH CODER FOR MULTIMEDIA APPLICATIONS

3.1 Introduction

This chapter forms the core of this thesis work. In this chapter we give a comprehensive analysis of the proposed coder. In section 3.2 an overview of the coder is given, highliting its main features. In section 3.3 the Encoder and its features are briefly introduced. A detailed presentation of the main features of the encoder is dealt in section 3.4 . The decoder and its main features are discussed in sections 3.5 and 3.6

3.2 Overview of the Proposed Coder

The coder is based on the code-excited linear-predictive (CELP) coding model. It operates on speech frames of 10 ms corresponding to 110 samples at a sampling rate of 11000 samples per second. For every 10 ms speech frame, the speech signal is analyzed to extract the parameters of the CELP model (linear-prediction filter coefficients, adaptive and fixed-codebook indices and gains). These parameters are encoded and transmitted. At the decoder, these parameters are used to retrieve the excitation and synthesis filter parameters. The speech is reconstructed by filtering this excitation through the short-term synthesis filter. The short-term synthesis filter is a 12th order linear prediction (LP) filter. The long-term, or pitch synthesis filter is implemented using the so-called adaptive-codebook approach. After computing the reconstructed speech, it is further enhanced by a postfilter. The coder has a total arithmetic delay of 15 msec resulting from the 10 msec speech frame and 5 msec look ahead.

3.3 Encoder

The encoding principle is shown in Fig. 3.1. The input speech signal is scaled in the preprocessing block. The scaling factor is chosen to be half in order to avoid overflow in case of fixed point implementation. This signal serves as the input signal for all subsequent analysis.

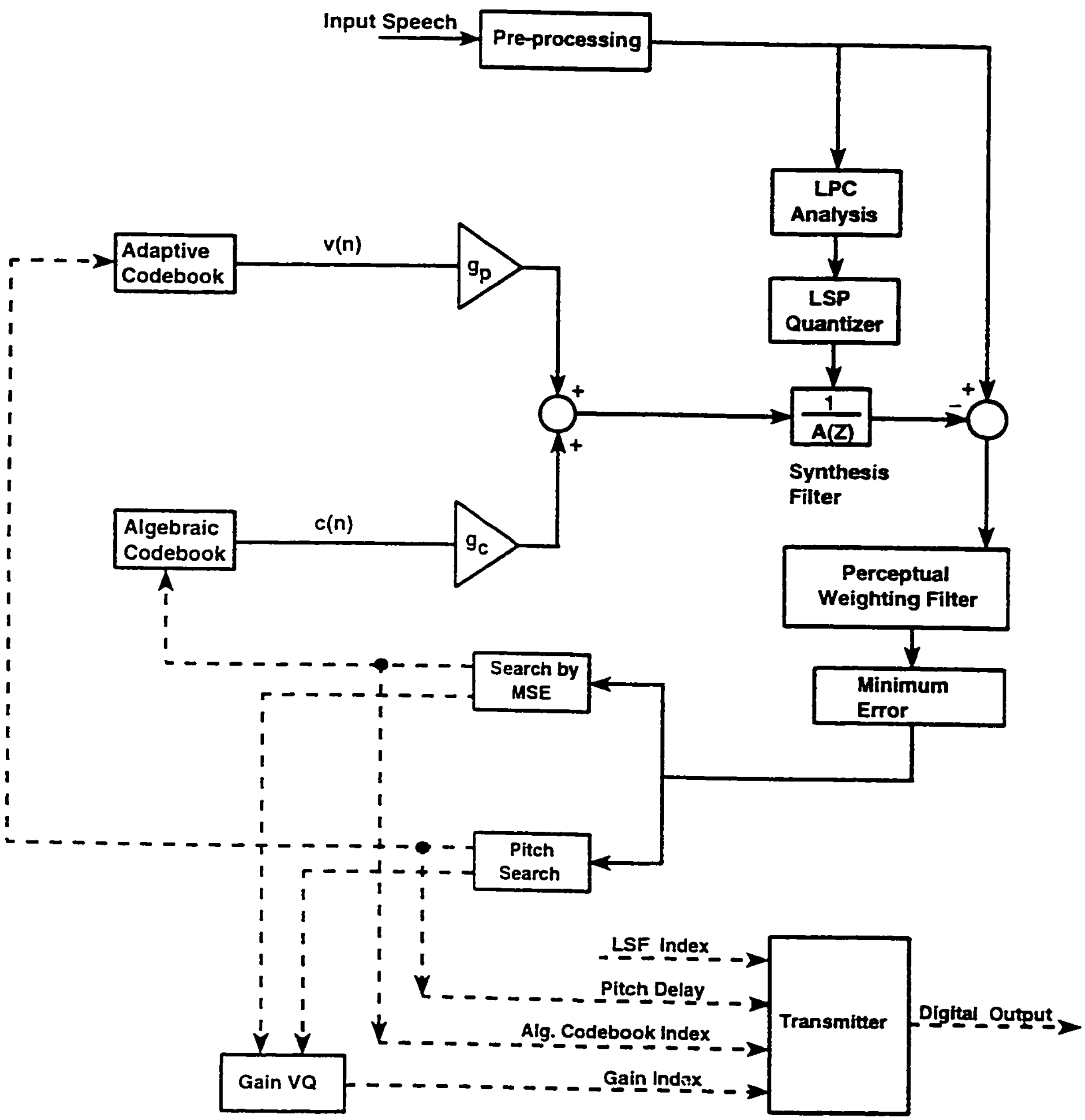


Figure 3.1: Encoder

Linear prediction analysis is done once per speech frame to compute the LP filter coefficients. These coefficients are converted into line spectral frequencies (LSFs) and are quantized using predictive split vector quantization(VQ). The short-term synthesis filter is a linear prediction filter. The excitation signal is chosen by using an analysis-by-synthesis search procedure in which the error between the original and reconstructed speech is minimized according to a perceptually weighted distortion measure. This is done by filtering the error signal with a perceptual weighting filter, whose coefficients are derived from unquantized LP filter. The amount of perceptual weighting is however fixed [34].

The speech frame is divided into two subframes and the excitation parameters are determined for each subframe. The quantized and unquantized LP filter coefficients are used for the second subframe, whereas in the first subframe interpolated LP filter coefficients are used. The interpolation is done in the LSF domain. The open-loop pitch delay is estimated once per each subframe, based on the perceptually weighted speech signal. This is done by filtering the error signal with a perceptual weighting filter, whose coefficients are derived from the unquantized LP filter. The LP residual signal is passed through this perceptual weighing synthesis filter $\frac{W(z)}{A(z)}$ to give the target signal $t(n)$. The initial states of these filters are updated by filtering the error signal between LP residual and excitation. This is equivalent to the common approach of subtracting the zero-input response of the weighted synthesis filter from the weighted speech signal. The impulse response $h(n)$ of the weighted synthesis filter is computed. Closed loop speech analysis is then done using the target signal and the impulse response, by searching around the value of the open-loop pitch delay. A fractional pitch delay of $\frac{1}{2}$ is made. The pitch delay is encoded with 8 bits in the first subframe and 5 bits in the second subframe. An algebraic codebook with 20 bits including 4 sign and 16 position bits is used for the fixed-codebook excitation. The gains of the adaptive and fixed-codebook contributions are vector quantized using 7 bits. Moving average prediction with fixed coefficients is applied to the fixed codebook gain [35].

3.4 Functional Description of the Encoder

In this section the different functions of the encoder are described. Speech is first sampled and then analyzed in frames of length 10 ms, corresponding to 110 samples each.

3.4.1 Preprocessing

The input signal is scaled down by a factor of 2 to reduce the possibility of overflows. No high pass filter is used in this coder indicating that frequencies as low as 50 Hz are also used in the analysis.

3.4.2 Linear Prediction Analysis

The short term analysis and synthesis filter are based on a 12th (M) order linear prediction (LP) filters. The speech is passed through a 30 msec hamming window prior to the linear prediction analysis. The LP analysis filter is given by

$$A(z) = 1 + \sum_{i=1}^{M} a_i z^{-i}, \tag{3.1}$$

and the synthesis filter is given by $H(z) = 1/A(z)$, where-in quantized linear prediction coefficients are used. Short-term prediction, or linear prediction analysis is performed once per speech frame using autocorrelation method. The autocorrelation coefficients of windowed speech are computed and converted to LP coefficients using the Levinson's algorithm. Then the LP coefficients are transformed to LSF for quantization and interpolation. The interpolated quantized and unquantized LSFs are converted back to the LP filter coefficients (to construct the synthesis and weighing filters for each subframe).

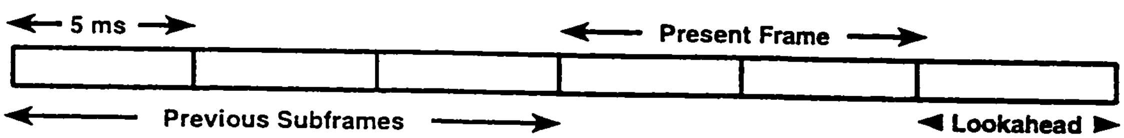


Figure 3.2: LP Analysis Window

The LP analysis window shown in Fig. 3.2 is given by

$$w(n) = .54 - .46\cos(\frac{2\pi n}{329}), \quad \text{n=0, ..., 329.} \tag{3.2}$$

The windowed speech signal

$$s'(n) = w(n) * s(n) \quad \text{n=0,..., 329,} \tag{3.3}$$

is used to compute the autocorrelation coefficients given by

$$r(k) = \sum_{n=k}^{329} s'(n)s'(n-k) \quad \text{k=0,..., M.} \tag{3.4}$$

The autocorrelation coefficients are used to obtain the LP filter coefficients a_i, $i = 1, ..., M$ by solving the set of equations

$$\sum_{i=1}^{M} a_i(n)r(|n-k|) = -r(k), \qquad k = 1, ..., M. \tag{3.5}$$

The set of equations in Eq.(3.5) are solved using the Levinson-Durbin recursion.

3.4.3 LP to LSF Conversion

Line Spectral Frequencies play an important role in speech coding as an alternative representation of Linear Prediction Coefficients. An efficient algorithm has been proposed for their efficient computation as part of this thesis work. However as a whole chapter has been dedicated to the LSFs we will only give a slight flavor about them in this section to keep in tune with our understanding of the coder.

The LSFs are the roots of the two polynomials P(z) and Q(z), given by

$$P(z) = A(z) + z^{-(M+1)}A(z^{-1}), \tag{3.6}$$

$$Q(z) = A(z) - z^{-(M+1)}A(z^{-1}), \tag{3.7}$$

where $A(z)$ is the forward LP analysis filter. These two polynomials after removing the roots at $z = -1$ and $z = +1$ can be written in the form

$$P'(z) = \Pi_{i=1,3,\ldots}^{M-1}(1 - 2cos\omega_i z^{-1} + z^{-2}), \tag{3.8}$$

$$Q'(z) = \Pi_{i=2,4,\ldots}^{M}(1 - 2cos\omega_i z^{-1} + z^{-2}), \tag{3.9}$$

where the roots of these polynomials ω_i are called the Line Spectral Frequencies.

The LP coefficients are converted to LSFs as the roots of Eqs. 3.8, 3.9 for interpolation and quantization purposes. Before conversion a bandwidth expansion of 15Hz is applied to the LP coefficients. This is done by multiplying the coefficients a_i by $(.9883)^i$.

3.4.4 LSF Quantization

Though scalar quantization of LSFs is still being used for high bit rate speech coding, vector quantization offers a considerable decrease in the total number of bits to be transmitted. Since the frame length is small, the redundancy arising from correlation between consecutive frames can be exploited. The LSF coefficients in this coder are quantized using predictive split vector quantization. This quantization is done in the following way:

1. The Long term DC component ω_{dc} is removed from the LSF vector ω' and is denoted by ω. ω_{dc} is computed over a training data of 100,000 LSF vectors as the minimum value of each component in the LSF vector.

2. A first order fixed predictor, (b=12/32), is applied to the previously quantized LSF vector $\tilde{\omega}_{n-1}$, to obtain the DC removed LSF vector $\hat{\omega}_n$, and the residual error vector, ω_r at frame n.

$$\begin{aligned}
\omega_n &= [\omega_{1,n}\ \omega_{2,n}\ \ldots\ \omega_{12,n}] \\
\hat{\omega}_n &= [\hat{\omega}_{1,n}\ \hat{\omega}_{2,n}\ \ldots\ \hat{\omega}_{12,n}] \\
\hat{\omega}_n &= b[\tilde{\omega}_{n-1} - \omega_{dc}] \\
\omega_r &= \omega_n - \hat{\omega}_n \qquad (3.10)
\end{aligned}$$

3. The residual LSF vector ω_r is divided into 3 subvectors with dimensions 4 each. Each of these subvector is vector quantized using a 7 bit codebook. The vector quantization is done using the LBG algorithm. This algorithm is discussed in the next section. The index l of the appropriate subvector code-book that minimizes the error criterion $E_{l,m}$ is selected.

$$E_{l,m} = (\omega'_m - \hat{\omega}_{l,m})' W_m (\omega'_m - \hat{\omega}_{l,m}), \quad 0 \leq m \leq 2, \quad 1 \leq l \leq 128 \tag{3.11}$$

where W_m is a diagonal weighting matrix, determined from the unquantized LSF vector ω' as [27]

$$\begin{aligned} w_{i,i} &= \frac{1}{min\{\omega'_i - \omega'_{i-1}, \omega'_{i+1} - \omega'_i\}}, \quad 2 \leq i \leq 11 \\ w_{1,1} &= \frac{1}{\omega'_2 - \omega'_1} \\ w_{12,12} &= \frac{1}{\omega'_{12} - \omega'_{11}} \end{aligned} \tag{3.12}$$

the idea is to give more weight to the LSF coefficient that is close to another coefficient.

4. At the decoder, first the three subvectors, ω_r are decoded to form a 12 th order vector. To this vector the predicted vector, $\hat{\omega}_{l,m}$ and the DC vector are added

to obtain the decoded LSF vector, $\tilde{\omega}_n$.

$$\hat{\omega}_n = b\tilde{\omega}_{n-1}$$

$$\tilde{\omega}_n = \omega_r + \hat{\omega}_n. \tag{3.13}$$

Vector Quantizer Design

Vector quantizer for the LSF parameters are designed using the LBG algorithm [16]. The basic technique is based on binary splitting. Firstly, for the whole set of training data a centroid based on the minimum distance is computed. Then this centroid is split into two centroids. After splitting, the training data is split into two partition of vectors, each set of which is closest in a minimum distance sense to the two centroids. Thereafter, the centroids of these two groups of vectors is computed. Then, these centroids are again each split into two and their partitions computed, followed by computing the centroids of these partitions. The above mentioned steps are repeated until the number of centroids is equal to the number of levels required in the quantizer. These centroids define the N-level quantizer. Following is an outline of the Linde Buzo and Gray (LBG) algorithm that is frequently used in our analysis. Flowchart for the algorithm is shown in Fig. 3.4.

The input to the quantizer is a set of M-dimensional line spectral frequencies

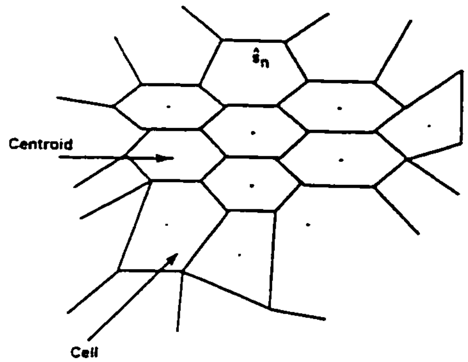


Figure 3.3: Cells for two dimensional VQ

(ω). The n-dimensional training sequence is given by $S = \{\Omega_j, j = 0, 1, ..., n-1\}$, where each Ω_j is the M-dimensional vector ω. The centroid of this training sequence is the vector u minimizing

$$\sum_{j:\Omega_j \in S} (\Omega_j - u)R(\Omega_j)(\Omega_j - u)'$$

and is found to be equivalent to

$$\hat{\omega}(S) = \{ \sum_{j:\Omega_j \in S} R(\Omega_j)\}^{-1} \sum_{j:\Omega_j \in S} R(\Omega_j)\Omega_j' \qquad (3.14)$$

where R is a matrix of autocorrelation values in a symmetric Toeplitz form. Each element of this matrix i.e., $R(i, j) = \beta(i-j)$ where

$$\beta(k) = \sum_{l=0}^{M-1-k} \omega(l)\omega(l-k) \qquad (3.15)$$

Distortion measure of Itakura [36] which is used here has the form

$$d(\omega, \hat{\omega}) = (\omega - \hat{\omega})R(\omega)(\omega - \hat{\omega})' \tag{3.16}$$

and $D(\Omega, \hat{\Omega})$ is the sum of this distortion computed over two groups of vectors Ω and $\hat{\Omega}$.

1. Initialization: Fix $N = 2^r$, r an integer, where N is the largest number of levels desired. M is the order of the LSF vector, $\epsilon = .005$. Set K=1.

   Given the training sequence $\Omega_j; j = 0, 1, \ldots, n-1$, set $A = \Omega_j; j = 0, 1, \ldots, n-1$, the training sequence alphabet. Define $\hat{A}(1) = \hat{t}(A)$, the centroid of the entire training sequence using Eq. (3.14).

2. Splitting: Given $\hat{A}(K) = \{Y_i, i = 1, \ldots, K\}$, where each Y_i is a M-dimensional vector y. Split each reproduction vector Y_i into $Y_i + \epsilon$ and $Y_i - \epsilon$ where ϵ is the fixed perturbation vector. Set $\hat{A}_0(2K) = \{Y_i + \epsilon, Y_i - \epsilon, i = 1, \ldots, K\}$ and then replace K by 2K.

3. Set $m = 0$ and $D_{-1} = \infty$

4. Given $\hat{A}_m(K) = \{Y_1, \ldots Y_K\}$, find its optimum partition $P(\hat{A}_m(K)) = \{S_i, i = 1, \ldots, Y_K\}$, that is $\Omega_j \in S_i$ if $d(\Omega_j, Y_i) \leq d((\Omega_j, Y_l)$, for all l. The optimal partition is the set of Ω_j which can be represented by Y_i. Compute the resulting

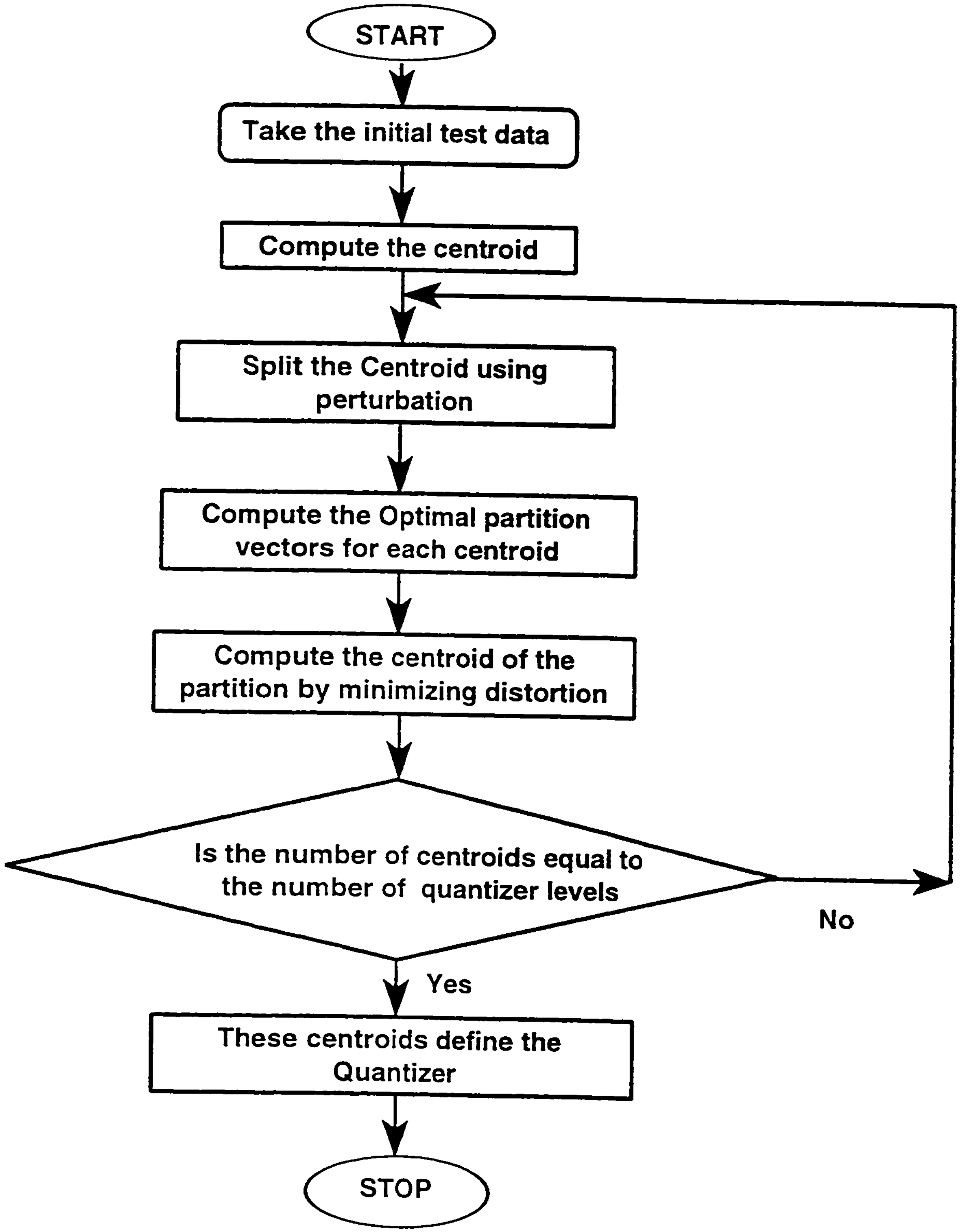


Figure 3.4: Vector Quantizer Algorithm

distortion

$$D_m = D(\{\hat{A}_m(K), P(\hat{A}_m(K))\})$$

$$= n^{-1} \sum_{j=0}^{n-1} \min_{Y \in \hat{A}_m} d(\Omega_j, Y)$$

5. If $(D_{m-1} - D_m)/D_m \leq \epsilon = .005$, then go to step (7), otherwise continue.

6. Find the optimal reproduction alphabet $\hat{A}_{m+1}(K) = \hat{t}(P(\hat{A}_m(K))) = \{\hat{t}(S_i): i = 1, ...K\}$ for $P(\hat{A}_m(K))$. Replace m by $m+1$ and go to (4).

7. Set $\hat{A}(K) = \hat{A}_m(K)$. The final K-level quantizer is described by $\hat{A}(m)$. If $K = N$, halt with the final quantizer described by $\hat{A}(N)$, otherwise go to step(2).

LSF Interpolation

The quantized LP coefficients are used for the second subframe. For the first subframe, the quantized LP coefficients are obtained by linear interpolation of the corresponding parameters in the adjacent subframes. The interpolation is done in cosine domain i.e., on the LSPs, where LSP =cos (LSF). Let $x_i^{present}$ be the LSP coefficient for the current 10 ms frame, and $x_i^{previous}$ be the LSP coefficient computed in the previous 10 ms frame. The interpolated LSP coefficients in each of the two subframes are given by:

$$\text{subframe 1} : x_i^{1} = .7x_i^{present} + .3x_i^{previous}, \quad i=1.....12$$

$$\text{subframe 2} : x_i^{2} = x_i^{present}, \quad i=1.....12 \qquad (3.17)$$

3.4.5 LSF to LP Conversion

Once the LSF coefficients are quantized and interpolated, they are converted back to the LP coefficients a_i. This conversion is done as follows. First the coefficients of P'(z) and Q'(z) are found. These are then multiplied by $1 + z^{-1}$ and $1 - z^{-1}$ to obtain the coefficients of P(z) and Q(z). Finally the coefficients of A(z) are found using

$$A(z) = \frac{P(z) + Q(z)}{2} \qquad (3.18)$$

Firstly, the coefficients of P'(z) are found using the recursion

$for \;\; i = 1 \;\; to \;\; 6$

$\quad p'(i) = -2x_{2i-1}p'(i-1) + 2p'(i-2)$

$\quad for \;\; j = i-1 \;\; down \; to \;\; 1$

$\quad\quad p'_i(j) = p'_{i-1}(j) - 2x_{2i-1}p'_{i-1}(j-1) + p'_{i-1}(j-2)$

$\quad end$

end

with initial values of $p'(1) = 1$ and $p'(-1) = 0$. Similarly the coefficients q'(i) are obtained by replacing x_{2i-1} by x_{2i}. Thereafter the coefficients of P(z) and Q(z) are obtained as

$$p(i) = p'(i) + p'(i-1), \quad \text{for i=1, ..., 12}$$

$$q(i) = q'(i) + q'(i-1), \quad \text{for i=1, ..., 12} \tag{3.19}$$

and the LP coefficients are given by

$$a_i = .5 * p(i) + .5 * q(i), \quad \text{for i=1, ..., 12} \tag{3.20}$$

with $a_0 = 1$.

3.4.6 Perceptual Weighting Filter

The encoder uses an analysis-by-synthesis technique to determine the pitch and excitation codebook parameters. In this technique, the synthesized speech is computed for all candidate innovation sequences retaining the particular sequence that produces the output closest to the original sequence according to a perceptually weighted distortion measure. The error signal is filtered through a perceptual weighting filter which de-emphasizes the error at the formant regions of the speech spectrum. The perceptual weighting filter is based on unquantized LP filter coefficients

a_i and is given by

$$W(z) = \frac{A(z/\gamma_1)}{A(z/\gamma_2)} \tag{3.21}$$

where A(z) is the LP analysis filter. The value of γ_1 and γ_2 determine the amount of noise masking(no masking when $\gamma_1 = 0$ and $\gamma_2 = 1$, and full masking when $\gamma_1 = 1$ and $\gamma_2 = 0$). By properly adjusting these variables, we can make the weighting more effective. It has been found that some spectra require significant noise masking whereas others require almost no masking [24]. If the energy is concentrated at low frequencies then the level of noise masking must be increased to avoid unmasked noise at high and even medium frequencies. In this coder the value of γ_1 is 0.94 and that of γ_2 is 0.6.

3.4.7 Adaptive Codebook Search

The adaptive codebook parameters are the delay and the gain of the pitch filter. A two stage approach is used in the pitch analysis[30]. First an open loop pitch is estimated for the whole speech frame. Around this open loop pitch a closed loop pitch is found for both the subframes.

Open Loop Pitch

The open loop pitch is estimated using the weighted speech signal s_w. This weighted speech signal is the signal passed through the perceptual weighing filter of Eq. 3.21.

In the first step, 3 maxima of the absolute value of the correlation

$$X(k) = \sum_{n=0}^{109} s_w(n)s_w(n-k) \quad k = 27, ..., 192 \tag{3.22}$$

are found in the three ranges, covering the present and the previous frame. These ranges are

$$i = 1 \;\; : \;\; 110, ..., 192$$

$$i = 2 \;\; : \;\; 56, ..., 109$$

$$i = 3 \;\; : \;\; 27, ..., 55$$

The retained maxima $X(t_i)$, $i = 1, 2, 3$ is normalized as $X'(t_i)$.

$$X'(t_i) = \frac{X(t_i)}{\sqrt{\sum_{n=0}^{n=109} s_w^2(n-t_i)}}. \tag{3.23}$$

The winner among the three normalized correlations is selected by favouring the delays in the lower ranges. That is an index t_i is selected if $X'(t_i) > .85X'(t_{i+1})$. This is done to avoid pitch multiples.

Closed Loop Pitch

Having found the open-loop pitch T_{op}, a closed-loop pitch analysis is done around the open-loop pitch analysis on a subframe basis. In the first sub-frame, a fractional pitch delay T_1 is used with a resolution of 1/2 in the range $[27\,\frac{1}{2}, 117\,\frac{1}{2}]$ and integers in the range [118,192]. For the second subframe, a delay T_2 with a resolution of $\frac{1}{2}$ in the range $[\ int(T_1) - 7,\ int(T_1) + 7\frac{1}{2}]$ is computed , where $int(T_1)$ is the integer part of the fractional pitch delay T_1 of the first subframe.

In the first subframe the delay T_1 is found by searching a small range (8 samples) of delay values around the open-loop delay T_{op}. The search boundaries t_{min} and t_{max} are defined by

$$t_{min} = T_{op} - 4$$

$$if\ \ t_{min}\ <\ 28\ \ then$$

$$t_{min} = 28$$

$$t_{max} = t_{min} + 8$$

$$if\ \ t_{max} > 192\ \ then$$

$$t_{max} = 192$$

$$t_{min} = t_{max} - 8$$

$$end$$

For the second subframe, closed-loop analysis is done around the pitch selected in the first subframe to find the optimal delay T_2. The search boundaries are between $t_{min} - \frac{1}{2}$ and $t_{max} + \frac{1}{2}$, where t_{min} and t_{max} are derived from T_1 as follows:

$$t_{min} = int(T_1) - 7$$

$$if\ \ t_{min}\ <\ 28\ \ then$$

$$t_{min} = 28$$

$$t_{max} = t_{min} + 15$$

$if \quad t_{max} > 192 \quad then$

$$t_{max} = 192$$

$$t_{min} = t_{max} - 15$$

end

For each of the two subframes the optimal delay is determined by minimizing the mean-square weighted error between the original and synthesized speech. This is achieved by maximizing the term [37]

$$\tau_k = \frac{|\sum_{n=0}^{54} t(n) y_k(n)|}{\sqrt{\sum_{n=0}^{54} y_k(n) y_k(n)}} \tag{3.24}$$

where

1. $t(n)$ is the so called target signal given by subtracting the zero-input response of the weighted synthesis filter $H(z)W(z) = A(z/\gamma_1)/[\hat{A}(z)A(z/\gamma_2)]$ i.e., s_0 from the weighted input speech $s_w(n)$ and computed as

$$t(n) = s_w(n) - s_0(n). \tag{3.25}$$

2. $y_k(n)$ is the past filtered excitation at delay k. I.e., past excitation $u(n-k)$

convolved with the impulse response of the weighted synthesis filter $H(z)W(z)$ and given by

$$y_k(n) = u(n-k) * h(n). \tag{3.26}$$

The impulse response $h(n)$ is computed for each subframe by filtering a signal consisting of the coefficients of $A(z/\gamma_1)$ extended by zeros through the two filters $1/\hat{A}(z)$ and $A(z/\gamma_2)$.

Derivation of Eq. 3.24 is done in Appendix D. Note that the search range is limited around a preselected value, which is the open loop pitch T_{op} for the first subframe and T_1 for the second subframe. Once the integer pitch delay is determined, the fractional pitch, around that integer are tested. The fractional pitch search is performed by interpolating the normalized correlations in Eq. (3.24) and searching for their maximum. Interpolation is done using truncated hamming windowed sinc functions with 12-points(N).

$$w(k) = .54 + .46\cos(\frac{k\pi}{2N}) \tag{3.27}$$

$$W_f(j) = w(4(j+f))\frac{sin((j+f)\pi)}{(j+f)\pi}, \qquad \begin{array}{l} \text{where } , f = 1/2, \\ j = -N/2, -N/2+1, ..., N/2-1. \end{array} \tag{3.28}$$

Using the above coefficients the term in Eq. 3.24 is interpolated for a fraction

$f = 1/2$ as

$$\tau_f(i) = \sum_{j=-N/2}^{N/2-1} W_f(j)\tau(i - T + j), \quad \text{where } i = 0, ...54, \ f = 1/2. \tag{3.29}$$

Fractions i.e., +1/2 and −1/2 around the integer pitch (T)are tested. Fraction of −1/2 refers to an integer pitch of (T-1) and a fraction of +1/2. A 30-point hamming windowed sinc function is plotted in Fig. 3.5.

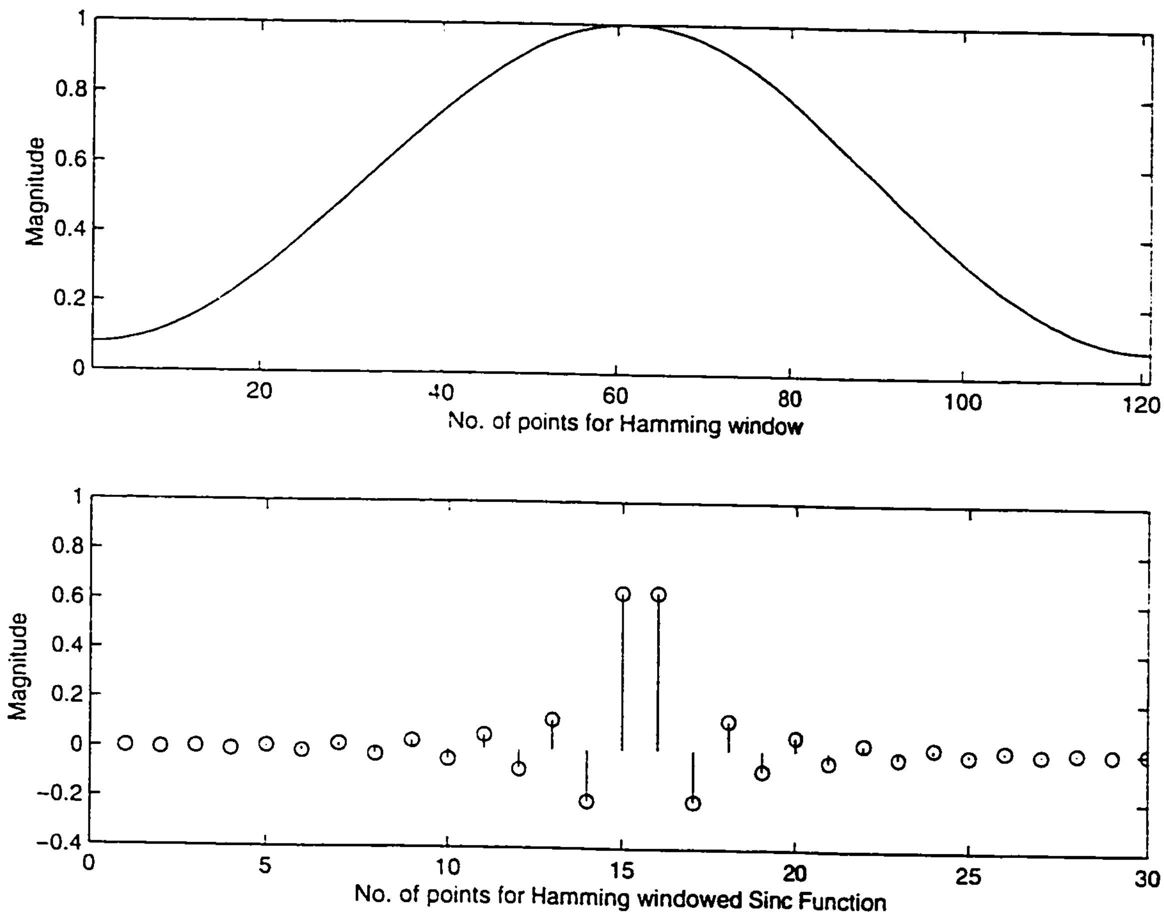


Figure 3.5: Hamming Windowed sinc function for interpolation

Adaptive Codebook Vector

Once the noninteger pitch is determined, the adaptive codebook vector $v(n)$ is computed by interpolating the past excitation signal $u(n)$. The noninteger delay consists of an integer part T and a fractional part f. The fractional part of the delay determines which set of weights is used to form the interpolated codeword. A recursive interpolating formula using hamming windowed sinc function of Eq. 3.28 using thirty points, to calculate the adaptive codeword v at delay $T+f$ is:

$$v(i) = \sum_{j=-N/2}^{N/2-1} W_f(j)u(i-T+j), \quad \text{where, } i = 0, ..., 54, \ f = 1/2, 0. \tag{3.30}$$

Codeword Computation for Adaptive-Codebook Delays

The pitch delay T, is encoded with 8 bits in the first subframe and the relative delay in the second subframe is encoded with 5 bits. A fractional delay T is represented by its integer part int(T), and a fractional part, $frac = 0, 1$. $frac = 1$ corresponds to $\frac{1}{2}$.

The pitch index P_1 is now encoded as

$$P_1 = \begin{cases} 2(int(T_1) - 27) + frac - 1 & if \ \ T_1 = [27, ..., 117], \quad frac = [0, 1] \\ (int(T_1) - 117) + 180, & if \ \ T_1 = [118, ..., 192], \quad frac = 0. \end{cases}$$

The value of the pitch delay T_2 is encoded relative to the value of T_1. Using the same interpretation as before, the fractional delay T_2 represented by its integer part $int(T_2)$, and a fractional part $frac = [0, 1]$ is encoded as

$$P_2 = 2(int(T_2) - t_{min}) + frac, \tag{3.31}$$

where t_{min}, is derived earlier.

Computation of the Adaptive-Codebook Gain

Once the adaptive-codebook delay is determined, the adaptive-codebook gain g_p is computed for each subframe as

$$g_p = \frac{\sum_{n=0}^{54} t(n)y(n)}{\sum_{n=0}^{54} y(n)y(n)} \tag{3.32}$$

where $y(n)$ is the adaptive codebook vector $v(n)$ of Eq. 3.30 convolved with $h(n)$, also called the filtered adaptive codebook vector. The gain is computed for each of the two subframes.

3.4.8 Algebraic Codebook: Structure and Search

The algebraic codebooks are used to populate the excitation codebooks. Algebraic codebooks are generally designed using interleaved single-pulse permutation designation. In this an excitation vector contains a few number of nonzero pulses with predefined interlaced sets of positions. The pulses have their amplitudes fixed to

+1, -1, or 0 (they could as well have other values), and each pulse have a set of possible positions distinct from the positions of the other pulses. The excitation code is identified by the position of its nonzero pulses. Thus, searching the codebook is in essence searching the optimum position of nonzero pulses. The codebook used in this coder is shown in Table. 3.1.

Pulse	Sign		Positions
i_0	s_0: $\pm$ 1	m_0:	0,4,8,12,16,20,24,28,32,36,40,44,48,52
i_1	s_1: $\pm$ 1	m_1:	1,5,9,13,17,21,25,29,33,37,41,45,49,53
i_2	s_2: $\pm$ 1	m_2:	2,6,10,14,18,22,26,30,34,38,42,46,50,54
i_3	s_3: $\pm$ 1	m_3:	3,7,11,15,19,23,27,31,35,39,43,47,51

Table 3.1: Structure of the fixed codebook

The codebook vector $c(n)$ is constructed by taking a zero vector of dimension 55, which is the number of samples in subframe and putting the 4 unit pulses at the found locations, multiplied with their corresponding sign:

$$c(n) = s_0\delta(n-m_0)+s_1\delta(n-m_1)+s_2\delta(n-m_2)+s_3\delta(n-m_3), \qquad n = 0,\ldots,54 \quad (3.33)$$

where $\delta(0)$ is a unit pulse.

The fixed codebook is searched by minimizing the mean-squared error between the weighted input speech $s_w(n)$ and the weighted reconstructed speech. The target signal used in the closed-loop pitch search is updated by subtracting the adaptive-codebook contribution. That is

$$t'(n) = t(n) - g_p y(n), \qquad n = 0, \ldots, 54. \tag{3.34}$$

where $y(n)$ is the filtered adaptive-codebook vector, and g_p is the adaptive-codebook gain of Eq. 3.32.

The matrix H is defined as the lower triangular Toeplitz convolution matrix with diagonal h(0) and lower diagonals h(1), ..., h(54). The matrix $\Phi = H'H$ contains the correlations of h(n), and the elements of this symmetric matrix are given by

$$\phi(i,j) = \sum_{n=j}^{54} h(n-i)h(n-j), \qquad i = 0, \ldots, 54, j = i, \ldots, 54. \tag{3.35}$$

The correlation signal d(n) is obtained from the target signal $t'(n)$ and the impulse response h(n)

$$d(n) = \sum_{i=n}^{54} t'(i)h(i-n), \qquad n = 0, \ldots, 54. \tag{3.36}$$

If c_k is the k-th fixed-codebook vector, then the codebook is searched by maximizing the term

$$\frac{C_k^2}{E_k} = \frac{(\sum_{n=0}^{54} d(n)c_k(n))^2}{c_k^t \Phi c_k} \tag{3.37}$$

The signal $d(n)$ and the matrix Φ are computed before the codebook search. Note that only the elements actually needed are computed and an efficient storage procedure of [38] has been used to speed up the search procedure.

The algebraic structure of the codebook allows for a fast search procedure since the codebook vector C_k contains only four nonzero pulses. The correlation in the numerator of Eq. 3.37 for a given vector C_k is given by

$$C = \sum_{i=0}^{3} s_i d(m_i) \tag{3.38}$$

where m_i is the position of the ith pulse and s_i is its amplitude. The energy in the denominator of Eq. 3.37 is given by

$$E = \sum_{i=0}^{3} \phi(m_i, m_i) + 2\sum_{i=0}^{2} \sum_{j=i+1}^{2} s_i s_j \phi(m_i, m_j). \tag{3.39}$$

To simplify the search procedure, the pulse amplitudes are predetermined by quantizing the signal $d(n)$. This is done by setting the amplitude of a pulse at a certain position equal to the sign of $d(n)$ at that position. Before the codebook search, the following steps are done. First, the signal $d(n)$ is decomposed into two parts: its absolute value $|d(n)|$ and its sign, $sign$ [d(n)]. Second, the matrix Φ is changed by including the sign information: that is,

$$\phi'(i,j) = sign[d(i)]sign[d(j)]\phi(i,j), \quad i = 0, ..., 54, \quad j = i+1, ..., 54 \tag{3.40}$$

The main-diagonal elements of Φ are scaled to remove the factor 2 in Eq. 3.39

$$\phi'(i,i) = 0.5\phi(i,i), \qquad i = 0, \ldots, 54. \tag{3.41}$$

The correlation in Eq. 3.38 is now given by

$$C = |d(m_0)| + |d(m_1)| + |d(m_2)| + |d(m_3)| \tag{3.42}$$

and the energy in Eq. 3.39 is given by

$$\begin{aligned} E/2 = {} & \phi'(m_0, m_0) + \\ & \phi'(m_0, m_1) + \phi'(m_1, m_1) + \\ & \phi'(m_0, m_2) + \phi'(m_1, m_2) + \phi'(m_2, m_2) + \\ & \phi'(m_0, m_3) + \phi'(m_1, m_3) + \phi'(m_2, m_3) + \phi'(m_3, m_3) \end{aligned} \tag{3.43}$$

The correlation requires 4(No. of pulses) additions and the energy term $\frac{4*(4+1)}{2}$ additions and one multiplication. However by changing only one pulse position at a time, updating the term in Eq. 3.37 becomes much simpler. The search is performed in 4 nested loops, where each loop corresponds to one pulse position. In each loop, the contribution of a new pulse is added. Thus, in the inner most loop, the contribution of the last pulse is added, so the correlation requires one addition and the energy 4 additions and one multiplication. If an exhaustive search is to be carried out, this approach is quite efficient. However, a focused search approach is used to further simplify the search procedure [38]. In this approach a precomputed threshold is tested before entering the last loop, and the loop is entered only if this threshold is exceeded. The maximum number of times the loop can be entered is fixed so that a low percentage of the codebook is searched. The threshold is computed based on maximum and average of the correlation C of the first three pulses. The maximum absolute correlation and the average correlation due to the contribution of the first three pulses, max_3 and av_3, are found before the codebook search. The threshold is given by

$$thr_3 = av_3 + K_3(max_3 - av_3) \tag{3.44}$$

The fourth loop is entered only if the absolute correlation (due to three pulses) exceeds thr_3, where $0 \leq K_3 < 1$. The value of K_3 controls the percentage of codebook search and it is set here to 0.4, this results in searching only 1.5% of the whole codebook [38]. Note that changing the value of K_3 results in a variable search time. A flow chart for this codebook search is shown in Fig. 3.6.

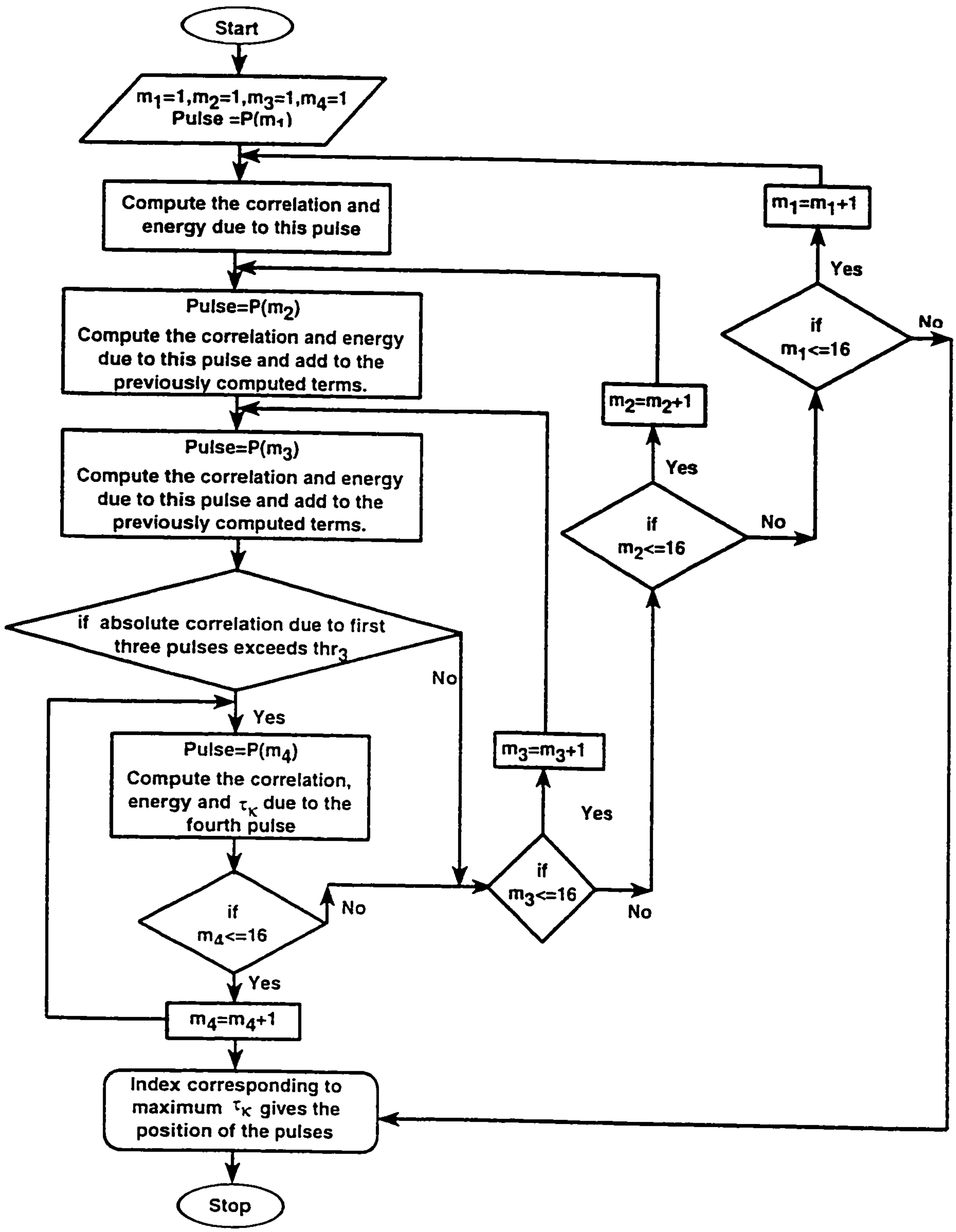


Figure 3.6: Flow chart for fixed codebook search

Codeword Computation of the Fixed Codebook

The pulse positions of the pulses i_0, i_1, i_2 and i_4 are encoded with 4 bits each. Each pulse amplitude is encoded with 1 bit. This gives a total of 20 bits

$$s = s_0 + 2s_1, +4s_2 + 8s_3, \tag{3.45}$$

and the fixed-codebook codeword is obtained from

$$C = (m_0/4) + 16(m_1/4) + 256(m_2/4) + 4096(m_3/4), \tag{3.46}$$

Computation of the Fixed-Codebook Gain

Once the fixed-codebook is determined, its gain g_c is computed as

$$g_c = \frac{\sum_{n=0}^{54} t'(n)c_z(n)}{\sum_{n=0}^{54} c_z(n)c_z(n)} \tag{3.47}$$

where $c_z(n)$ is the fixed codebook vector convolved with h(n) and given by

$$c_z(n) = \sum_{i=0}^{n} c(i)h(n-i) \qquad n = 0, ..., 54. \tag{3.48}$$

As the value of fixed codebook gain can have a large range moving average prediction is applied to this gain and a correction factor resulting from this prediction is transmitted. This gain prediction is discussed in the next section.

Fixed Codebook Gain Prediction

In this section we deal with computing the value of the correction factor γ, which is a parameter in the vector quantized gain codebook.

Preliminaries

The fixed-codebook gain g_c can be expressed as

$$g_c = \gamma g_c' \tag{3.49}$$

where g_c' is a predicted gain based on previous fixed-codebook energies, and γ is a correction factor. γ is the factor that is to be transmitted. The mean energy of the fixed-codebook contribution is given by

$$E = 10 \log \left(\frac{1}{55}\sum_{n=0}^{54} c(n)^2\right) \tag{3.50}$$

After scaling the vector $c(n)$ with the fixed-codebook gain g_c the energy of the scaled fixed codebook is given by $20 \log (g_c) + E$. Let $E^{(m)}$ be the mean-removed energy (in dB) of the (scaled) fixed-codebook contribution at subframe m, given by

$$E^{(m)} = 20 \log g_c + E - \overline{E}, \tag{3.51}$$

where $\overline{E} = 30dB$ is the mean energy of the fixed-codebook excitation. The gain g_c can be expressed

$$g_c = 10^{(E^{(m)}+\overline{E}-E)/20}. \tag{3.52}$$

The predicted gain g_c is found by predicting the log-energy of the current fixed-codebook

$$\tilde{E}^{(m)} = \sum_{i=1}^{4} b_i \hat{U}^{(m-i)}. \tag{3.53}$$

where $[b_1\ b_2\ b_3\ b_4] = [0.68\ 0.58\ 0.34\ 0.19]$ are the MA prediction coefficients, and $\hat{U}^{(m-i)}$ is the quantized version of the prediction error $U^{(m)}$ at subframe m, defined by

$$U^{(m)} = E^{(m)} - \tilde{E}^{(m)}. \tag{3.54}$$

The predicted gain g_c is found by replacing $E^{(m)}$ by its predicted value in Eq. 3.52.

$$g_c' = 10^{(\tilde{E}^{(m)}+\overline{E}-E)/20}. \tag{3.55}$$

The correction factor γ is related to the gain-prediction error by

$$U^{(m)} = E^{(m)} - \tilde{E}^{(m)} = 20\log(\gamma). \tag{3.56}$$

Main aspects in prediction

1. During the coding process the value of g_c' is computed for every subframe using Eq. 3.55

2. The value of γ is selected from the codebook by using Eq. 3.49 and minimizing the error in Eq. 3.57.

3.4.9 Gain Quantization

The adaptive-codebook gain (pitch gain) and the fixed codebook gain are vector quantized using

$$E = ||t - g_p y - g_c c_z||$$
$$= t't + g_p^2 y'y + g_c^2 c_z' c_z - 2g_p t'y - 2g_c t'c_z + 2g_p g_c y'c_z \tag{3.57}$$

where t is the target vector . y is the filtered adaptive-codebook vector and c_z is the fixed-codebook vector convolved with $h(n)$ given by 3.48,

The codebook is generated using the LBG algorithm [39] with 128 levels corresponding to 7 bits. Vector quantization is done with adaptive gain as the first entry in the codebook and the correction factor γ as the second entry in the codebook. The entry corresponding to the minimum error in Eq. 3.57 is selected and passed on to the decoder.

3.4.10 Bit Allocation of the Coder

The bit allocation of the coder parameters is shown in Table 3.2. 21 bits are used for vector quantized line spectral frequencies, which define the short term filter. The long term filter, characterized by the pitch delay uses 8 bits for the first subframe and differential coding of the second subframe uses 5 bits. Fixed codebook or the algebraic codebook uses 20 bits for each subframe including the sign and the position bits. The gains are represented by vector quantized codebook of 128 levels corresponding to 7 bits for each subframe. The total number of bits needed to represent each frame of 10ms is 88, this corresponds to 8800 bits per second, which is the bit rate of the coder.

Parameter	Codeword	Subframe 1	Subframe 2	Total/frame
Line Spectral Frequency	L_1, L_2, L_3			21
Adaptive-codebook delay	P_1, P_2	8	5	13
Fixed-codebook index	C_1, C_2	16	16	32
Fixed-codebook sign	S_1, S_2	4	4	8
Codebook gains	G_1, G_2	7	7	14
Total				88

Table 3.2: Bit allocation of the coder

3.5 Decoder

The decoder principle is shown in Fig.3.7. First, the parameter indices are extracted from the received bit-stream. These indices are decoded to obtain the coder parameters corresponding to the speech frame. These parameters are the LSF coefficients, the 2 fractional pitch delays, the two fixed-codebook vectors, and the two sets of adaptive and fixed-codebook gains. The LSF coefficients are interpolated and converted to LP filter coefficients for each subframe. Then, for each subframe the

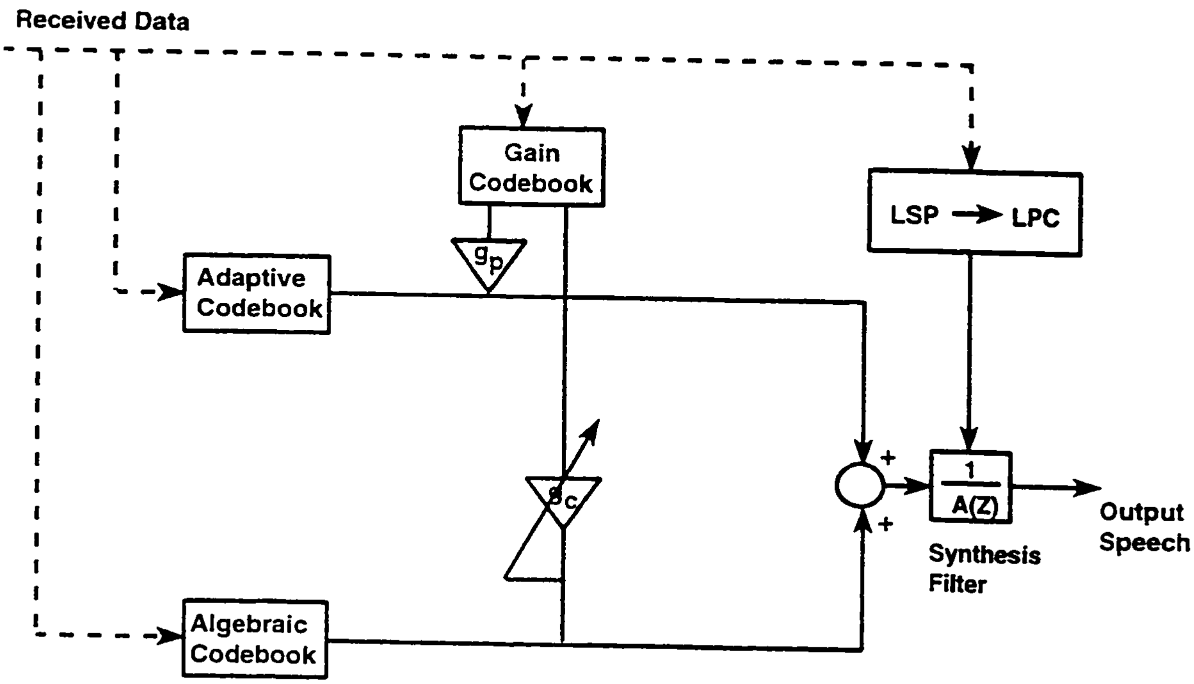


Figure 3.7: Decoder

excitation is constructed by adding the adaptive and fixed-codebook vectors scaled by their respective gains. The speech is reconstructed by filtering the excitation through the LP synthesis filter. The reconstructed speech signal is passed through a post-processing stage, which includes an adaptive post filter based on the long-term and short-term synthesis filters, followed by scaling operation.

3.6 Functional Description of the Decoder

The decoding procedure is just the complement of the encoding, further as in analysis-by-synthesis procedure decoder is a part of the encoder most of its quantitative aspects have been covered in the encoder. The decoding is done in the following order.

3.6.1 Decoding the LP Parameters

The received bit stream corresponding to the LSF parameters are decoded. The interpolation procedure described earlier is adapted. For each of the two subframes the interpolated LSF coefficients ω_i are converted to the LP coefficients a_i. This is done once for each frame. However the following steps have to be repeated for each subframe:

1. Decoding of pitch
2. Decoding of the fixed codebook vector
3. Decoding the gains
4. Reconstructing the speech

3.6.2 Decoding of Pitch

The adaptive codebook vector $v(n)$ is found by interpolating the past excitation $u(n)$ at the pitch delay. The index P_1 is used to find the integer and fractional part of the Pitch delay T_1. These are obtained as follows:

$if\ P_1 < 180$

$$int(T_1) = int((P_1 + 1)/2) + 27$$

$$frac = P_1 - 2 * int(T_1) + 55$$

$else$

$$int(T_1) = P_1 - 63$$

$$frac = 0$$

end

The integer and the fractional part of T_2 are obtained from P_2 and t_{min}, where t_{min} is derived from T_1 as:

$t_{min} = int(T_1) - 7$

$if\ \ t_{min}\ <\ 28\ \ then$

$$t_{min} = 28$$

$$t_{max} = t_{min} + 15$$

$if\ \ t_{max} > 192\ \ then$

$$t_{max} = 192$$

$$t_{min} = t_{max} - 15$$

end

Now T_2 is decoded using t_{min} as:

$$int(T_2) = int((P_2)/2) + t_{min}$$

$$frac = P_2 - 2 * (int(T2) - t_{min})$$

Having found the pitch delays, the adaptive codebook vector is given by the excitation vector at that delay. However, for fractional delay Eq. 3.30 is used to compute the adaptive codebook vector.

3.6.3 Decoding of Fixed-Codebook Vector

The code C is used to extract the positions of the pulses and S gives the sign of those pulses. Once the pulse position and their signs are obtained the fixed codebook vector is given by Eq. 3.33.

3.6.4 Decoding of the Gains

The fixed and the adaptive codebook gains are decoded from their corresponding bit streams as follows. First the index of the codebook is decoded to give the adaptive gain as the first vector and the correction factor γ as the second vector. The predicted fixed codebook gain given by Eq. 3.55 is multiplied to the correction factor to obtain the quantized fixed codebook gain.

3.6.5 Reconstruction of Speech

The excitation signal $u(n)$, which is the sum of adaptive codebook vector and the fixed-codebook vector multiplied by their respective gains; is passed through the LP synthesis filter to reconstruct the speech as

$$\hat{s}(n) = u(n) - \sum_{i=1}^{M} \hat{a}_i \hat{s}(n-i), \qquad n = 0, \ldots, 54. \tag{3.58}$$

3.6.6 Scaling

Finally the post processed speech is scaled up by a factor of two, this is to compensate for the down scaling at the encoder.

Chapter 4

EFFICIENT COMPUTATION OF THE LSFs

4.1 Introduction

Line Spectral Frequencies (LSFs) are an alternative representation of the Linear Prediction Coefficients. Quantization of LSFs offers good reduction in transmission bit-rate thereby increasing compression. Section 2.2 deals with the fundamentals of the LSFs. Starting from the Linear Prediction model, the derivation of LSFs is presented. Section 2.3 presents an algorithm for efficient computation of the LSFs. Issue like floating point implementation is also considered.

4.2 Line Spectral Frequencies

The concept of the line spectral frequencies LSFs ($0 < \omega < \pi$) was first introduced by Itakura [9]. These LSFs in the cosine domain i.e., $x = cos(\omega)$ are called the Line Spectrum Pairs LSPs. The LSFs have some useful and desirable properties which makes them an excellent choice for use in speech coding. Some of its properties are bounded range, inter-frame correlation and easy stability check. Further as the LSFs allow frame to frame interpolation they can be quantized effectively by taking into account their spectral features.

Computation of the LSFs was first done using Kabal's [40] method by finding the roots of the two polynomials $P'(x)$ and $Q'(x)$. In their numerical solution, zero crossings are searched starting at $x = +1$, with decrements of $\delta = .02$. Once a zero crossing is found, its position is refined by four successive bisections and a final linear interpolation. An efficient recursive evaluation requiring lesser computation is also proposed in [40].

In [41] a more efficient method with less precision is proposed. The computations were reduced by reducing the number of zero crossing search. In [21] the discrete cosine transform of the coefficients of $P'(x)$ and $Q'(x)$ are used to compute the LSPs. However, almost all techniques used the zero crossings of the two polynomials $P'(x)$ and $Q'(x)$ to compute the LSPs, as the zeros of these polynomials in closed form cannot be calculated.

The emphasis of this work was on the efficient computation of the line spectral frequencies without getting into the details of the zero crossings of the two polynomials. It involves an iterative root finding algorithm for the two polynomials. This technique is useful because most applications do not require that the LSFs be computed to extreme precision, thereby saving huge computational overhead in real-time environments.

4.2.1 Interpreting Line Spectral Frequencies

The speech production model in terms of the linear prediction coefficients a_i is given by

$$A(z) = 1 + \sum_{i=1}^{M} a_i z^{-i} \tag{4.1}$$

Expressing a new polynomial known as the backward polynomial, it can be represented as

$$B(z) = z^{-(M+1)} A(z^{-1}) \tag{4.2}$$

By expanding, the above two polynomials can be written as:

$$A(z) = 1 + a_1 z^{-1} + a_2 z^{-2} + ... + a_{M-1} z^{-(M-1)} + a_M z^{-(M)}$$

$$B(z) = a_M z^{-1} + a_{M-1} z^{-2} + ... + a_2 z^{-(M-1)} + a_1 z^{-M} + z^{-(M+1)} \tag{4.3}$$

Two new polynomials are then formed, corresponding to the closed and open glottis conditions:

$$P(z) = A(z) + B(z) \tag{4.4}$$

$$Q(z) = A(z) - B(z) \tag{4.5}$$

Each polynomial is of the order M+1, but P(z) always has a root at $z = -1$ while Q(z) has a root at $z = +1$. Hence the two can be factored into Mth order polynomials :

$$P'(z) = \frac{P(z)}{(1 + z^{-1})} \tag{4.6}$$

$$Q'(z) = \frac{Q(z)}{(1 - z^{-1})} \tag{4.7}$$

An important point is that all the zeros of $P'(z)$ and $Q'(z)$ lie on the unit circle. Hence for M even each can be expressed as the product of $M/2$ second order polynomials:

$$P'(z) = \Pi_{i=1,3,...}^{M-1}(1 - 2cos\omega_i z^{-1} + z^{-2}) \tag{4.8}$$

$$Q'(z) = \Pi_{i=2,4,...}^{M}(1 - 2cos\omega_i z^{-1} + z^{-2}) \tag{4.9}$$

where the ω_i are real valued and correspond to the angles of the root locations along the unit circle. The parameters ω_i are called Line Spectral Frequencies (LSF) [22].

They can be ordered such that

$$\omega_1 < \omega_3 < \ldots < \omega_{M-1} < \omega_{M+1}$$

$$\omega_0 < \omega_2 < \ldots < \omega_{M-2} < \omega_M$$

where $\omega_0 = 0$ and $\omega_{M+1} = \pi$. Another point is that if the original LP polynomial has all its roots inside the unit circle, the LSF, $\{\omega_i\}$ must be related by:

$$\omega_0 < \omega_1 < \omega_2 < \ldots < \omega_M < \omega_{M+1}$$

that is the roots of the two polynomials must be interlaced. This indicates that if a synthesis filter is stable, its coefficients can be represented as an ordered set of real numbers.

4.2.2 Computational Considerations

Due to the nature of the polynomials, $P(z)$ and $Q(z)$ it is possible to formulate simple and effective algorithm to find ω_i. In particular the fact that all the roots lie on the unit circle, and that the LSF have a natural order can be used to great advantage in formulating computationally efficient solutions. The most straighforward method of locating the roots is to compute the magnitude of each polynomial along on the unit circle:

$$|P(e^{j\omega})| = |1 + \sum_{k=1}^{M} P_k e^{jk\omega}| \tag{4.10}$$

searching for minima as ω is varied over the range $(0, \pi)$. The above equation can be reduced to a simpler form . When the root at $z = +1$ or $z = -1$ is factored out, the resulting polynomials $P'(z)$ and $Q'(z)$ are both symmetric in the form (for M even)

$$P'(z) = z^{-N}(p_N + \sum_{i=0}^{N-1} p_i[z^{(N-i)} + z^{-(N-i)}]). \tag{4.11}$$

where $N = M/2$, and p_i are the coefficients of $P(z)$. Evaluating on the unit circle this equation reduces to

$$P'(\omega) = 2e^{-j\omega N} P_c(\omega). \tag{4.12}$$

where,

$$P_c(\omega) = .5p_N + \sum_{i=0}^{N-1} p_i cos((N - i)\omega). \tag{4.13}$$

In order to locate the roots of $P'(z)$ it is necessary only to find the zeros of $P_c(\omega)$. The function $Q_c(\omega)$ is defined exactly as $P_c(\omega)$ with the coefficients of $Q'(z)$ replacing those of $P'(z)$.

During the initial stages of this thesis work the zeros of this polynomial were searched with ω varying from 0 to π in steps of $\delta\omega = \pi/512$. However more efficient techniques were developed for the computation of the LSFs and these are presented in this chapter.

The roots of Eq. 4.13 i.e., ω are called the LSFs. And these roots in cosine domain i.e., $x = cos(\omega)$ are the LSPs. We will now discuss two techniques for directly computing the LSFs and LSPs without getting into the details of the zero crossing. Firstly, the algorithm which is used in both the techniques is presented. The flow chart of the algorithm is illustrated in Fig. 4.1

4.3 Algorithm

The technique that is used in computing the LSFs is a simple iterative one which involves computing the roots of the polynomials $P_c(\omega)$ and $Q_c(\omega)$. While roots are computed as $x = cos(\omega)$ in the first technique, ω is computed directly in the second technique. A simple second order root finding formula like the Newton's method can be employed in both the techniques. The different steps in the algorithm are outlined below:

1. First we begin with an initial value of ω such as $.04\pi$. This corresponds to x=.9974. In general the root is treated as α.

2. For the value of i from 1 to N the following steps are repeated.

   (a) Any root finding formula like the Newton's is employed to locate the root of $P_c(\alpha)$. In a few iterations the root is computed which is α_{2*i-1}.

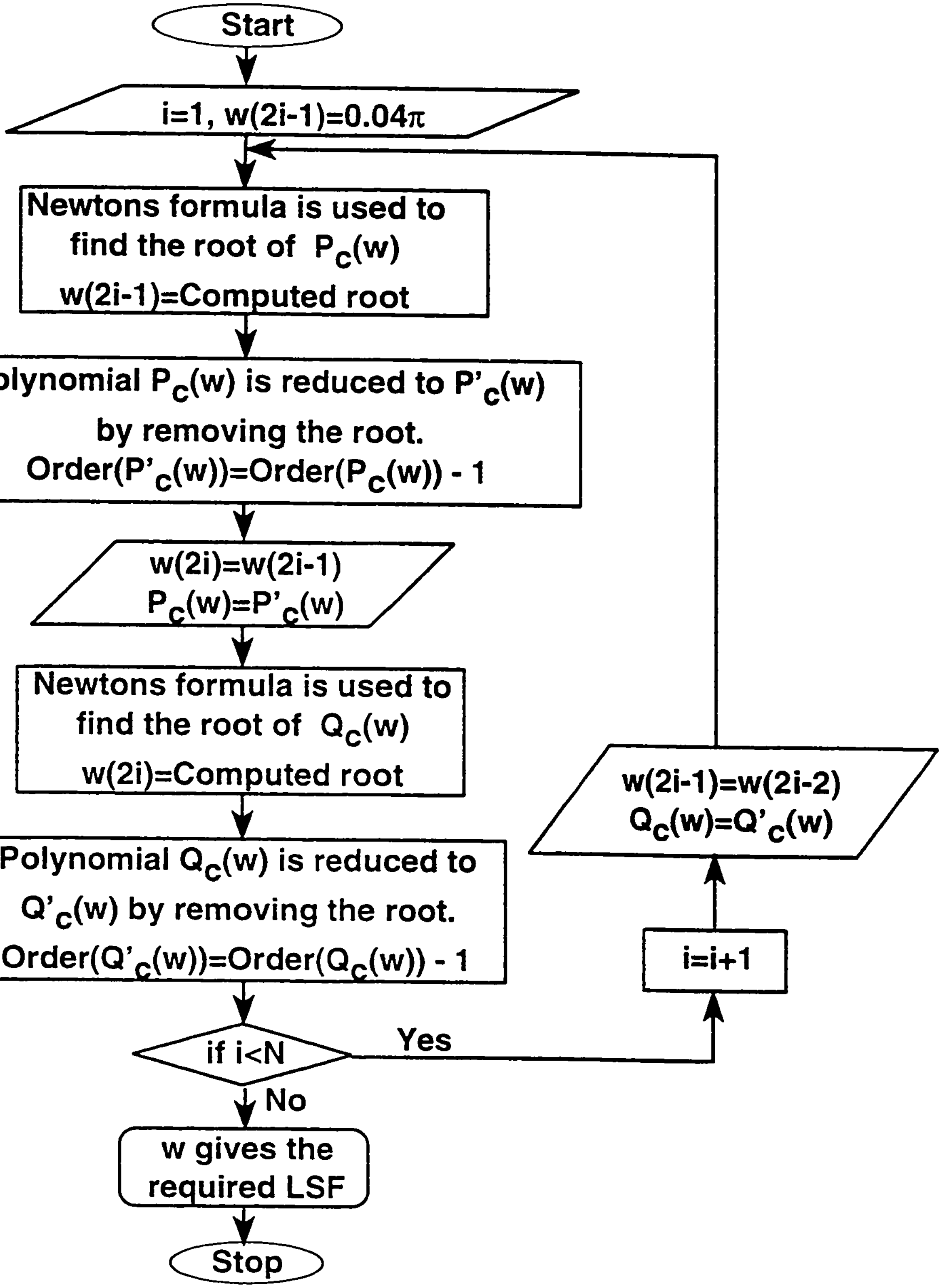


Figure 4.1: Flow chart for LSF computation algorithm

(b) Using α_{2*i-1} as a starting point in the root finding formula the next value i.e., α_{2*i} is found as the root of the polynomial $Q_c(\alpha)$.

(c) The value of the root found in the above steps is factored out from the above polynomials to obtain $N-i$th order polynomial.

(d) The root α_{2*i} is used as the starting point and returned to step (a) with increment in i.

3. The roots are checked to find if they are properly interlaced, if not then they are properly rearranged.

4. The roots computed above are either the line spectrum pairs using the first technique or the line spectral frequencies using the second technique. A look up table will help in finding one from the other.

4.3.1 Technique 1

Line spectrum pairs are computed as the roots of the equations $P_c(x)$ and $Q_c(x)$ with $x = cos(\omega)$. For a 10th order LPC polynomial $P_c(x)$ can be represented as a fifth order polynomial as

$$P_5 = x^5 + z_5(1)x^4 + z_5(2)x^3 + z_5(3)x^2 + z_5(4)x + z_5(5) \tag{4.14}$$

where z_5 are the coefficients of P_5. Similar equation can be used to represent Q_5. Detailed analysis is presented in Appendix A. The root of the above polynomial is computed as α_1 using Newton's formula (Eq. A.2). This root is then removed from the 5th order polynomial: the resulting fourth order polynomial is then represented as

$$P_4 = x^4 + z_4(1)x^3 + z_4(2)x^2 + z_4(3)x + z_4(4) \tag{4.15}$$

In general the polynomial P_c or Q_c is denoted as Z. The nth order polynomial and the coefficients after factorizing out the kth root is given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)x^{n-i} \tag{4.16}$$

where, $z_n(0) = 1$, $z_n(n) = \frac{-z_{n-1}(n+1)}{\alpha_k}$ and $z_n(j) = \frac{z_n(j+1) - z_{n+1}(j)}{\alpha_k}$ for $j = n-1, \ldots, 1$ and $n = N-1, \ldots, 1$.

While the first root of P_5 is computed with an initial value of $\alpha = cos(.04\pi)$, this root then serves as the starting value for the first root in Q_5. Once this root is computed it is used as the initial value in computing the root of P_4. This procedure is continued until all the roots are found.

4.3.2 Technique 2

Since Line Spectral Frequencies are directly required in most of the application rather than the LSPs, in this algorithm direct computation of the LSFs is done.

However, computation of one from the other can be done using a simple look up table of either cosine or cosine inverse terms. The same algorithm is once again used here, however the polynomials that are used in the computation are $P_c(\omega)$ and $Q_c(\omega)$. Further, computations involved in the previous technique make it difficult for implementation in fixed point algorithm. In this approach multiple order computations are not involved and major work is reduced by using lookup table of cosine values. A simple second order root finding formula like the Newton's method can again be applied here. Chebyshev polynomials are used to represent the polynomial $P_c(\omega)$ and $Q_c(\omega)$. They have an efficient recurrent relation

$$T_{i+1} = \frac{1}{2}T_i * T_1 - T_{i-1} \tag{4.17}$$

where $T_i = cos(i * \omega)$.

In general the nth ordered polynomial is represented in terms of the Chebyshev polynomial as

$$Z_n = \sum_{i=0}^{n} z_n(i)T_i \tag{4.18}$$

Firstly, from the fifth order polynomial the root of P is found. Then with this as the initial value, the root of Q is found. Once the first root is computed the order of the polynomial is reduced, the nth order polynomial and the coefficients after factorizing out the kth root are given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)T_i \tag{4.19}$$

where,

$$z_n(l-1) = z_{n+1}(l) + 2z_n(l)cos(\alpha_k) - z_n(l+1) \quad \text{for l=N-k,...,1.}$$

and

$$z_n(0) = z_n(0)/2$$

$$z_{n+1}(N-k) = 1$$

$$z_n(N-k) = 0$$

$$z_n(N-k-1) = 1$$

While the first root of P_5 is computed with an initial value of $\omega = .04\pi$, this root then serves as the starting value for the first root in Q_5. Once this root is computed it is used as the initial value in computing the root of P_4. This procedure is continued until all the roots are found. Finally, the roots are checked to find if they are interlaced. If not then they are properly rearranged.

4.3.3 Fixed Point Implementation

This section deals with the issue related to fixed point implementation of Technique 2. In order to implement the algorithm in fixed point we first need to get all the intermediate results to lie between 0 and 1. To begin with we must first normalize all the coefficients of the polynomials and normalize each iteration in the Newton's formula. Detailed analysis is presented in Appendix C.

Normalizing the coefficients

z_n represents the coefficients of the polynomial before normalization and $z'(n)$ represents the same coefficients after normalization. The nth order polynomial and the coefficients after factorizing out the kth root (k=n+1) are given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)T_i \tag{4.20}$$

where,

$$z_n(l-1) = z_{n+1}(l) + 2z_n(l)cos(\alpha_k) - z_n(l+1) \quad \text{for l=N-k,...,1.}$$

and

$$z_n(0) = z_n(0)/2$$

$$z_{n+1}(N-k) = 1$$

$$z_n(N-k)=0$$

$$z_n(N-k-1)=1$$

A general routine to get normalized coefficients is outlined below:

1. The first set of polynomials i.e $z_n(i),\ i=0,...,n-1$ are normalized to obtain the values between 0 and 1 and denoted by z'_n. Normalization is done by dividing the coefficients by their maximum i.e., $z'_n(i) = z_n(i)/max(z_n(i))\ i = 0,...,n-1$.

2. Thereafter to obtain the normalized coefficients of any order n the following iteration has to be used with $z'_{n+1} = z'_{n+1}/4$ to obtain $0 \leq z'_n \leq 1$.

$$z'_n(l-1) = z'_{n+1}(l) + 2z'_n(l)cos(\alpha_k) - z'_n(l+1) \quad \text{for l=N-k,...,1.}$$

and

$$z'_n(0) = z'_n(0)/2$$

$$z'_{n+1}(N-k) = 1$$

$$z'_n(N-k) = 0$$

$$z'_n(N-k-1) = 1$$

Normalizing the Newton's Iteration

The Newton's formula used in these algorithms has the following structure involving the chebyshev polynomial as:

$$x_{n+1} = x_n + \frac{\sum_{i=0}^{n} z_n'(i)cos(i\pi x_n)}{\sum_{i=1}^{n} z_n'(i)i\pi sin(i\pi x_n)} \tag{4.21}$$

since the value of $z_n'(i)$ is less than 1 and the sin term is always less than 1, the denominator can have a maximum value of $\sum_{i=1}^{n} i\pi$ and we have

$$\sum_{i=1}^{n} i\pi = n(n+1)\pi/2, \tag{4.22}$$

dividing the numerator and denominator by this term we get

$$x_{n+1} = x_n + \frac{\sum_{i=0}^{n} z_n'(i)cos(i\pi x_n)\frac{2}{n(n+1)\pi}}{\sum_{i=1}^{n} z_n'(i)\frac{2\ i}{n(n+1)}sin(i\pi x_n)} \tag{4.23}$$

all the terms in the above equation are less than one hence it can be easily implemented using fixed point.

Chapter 5

RESULTS AND DISCUSSIONS

5.1 Introduction

This chapter is dedicated to results and discussions. Firstly, the standard G.729 is compared with the proposed coder in section 5.2. Results and discussion of the listening tests are treated in this section. Validation of various quantization schemes implemented in this work is done in section 5.3. Efficiency of the algorithm for LSF computation are discussed in section 5.4. Finally, in section 5.5 possible implementation of the proposed coder over a dedicated signal processor is discussed.

5.2 Comparison of Standard G.729 and Proposed Coder

The ITU-T standard G.729 for mobile communications is compared to the proposed coder in this section. Though both these coders are based on CELP analysis by synthesis technique they differ greatly in various aspects. The main difference arises due to the bandwidth of the speech signal under consideration. While the standard G.729 is restricted to telephone bandlimited signals, our coder covers a wider range of frequencies. Further it was found that the predictive split vector quantization of standard G.723.1 fared better in performance than the predictive 2-stage vector quantization of G.729. This was the reason why this technique has been used in our coder. Computational overhead was further reduced by using fixed spectral shaping and ordinary vector quantization of the gains. As G.729 was developed mainly for mobile radio, parity bit was used to check for transmission error, the same is not needed in our case. A summary of these comparisons is shown in Table 5.1.

Standard G. 729	Proposed Coder
Applicable for narrow band applications (300-3400 Hz)	Applicable for medium band applications (20-5400 Hz)
Sampling frequency is 8ks/s	Sampling frequency is 11ks/s
Transmission bit-rate is 8kb/s	Transmission bit-rate is 8.8 kb/s
Coder delay is 15 msec	Coder delay is 15 msec
Speech frame of 10 msec corresponding to 80 samples/frame	Speech frame of 10 msec corresponding to 110 samples/frame
Each frame is further divided into 2 subframes	Each frame is further divided into 2 subframes
Adaptive spectral shaping is used	Spectral shaping is fixed
10th order LP analysis filter is used	12th order LP analysis filter is used
Pitch analysis with $\frac{1}{3}$ resolution is used	Pitch analysis with $\frac{1}{2}$ resolution is used
Value of the pitch varies from 20 to 143	Value of the pitch varies from 27 to 192
Predictive split vector quantization of LSF using 18 bits with split structure at the second stage is used	LSF split vector quantization with 21 bits is used.
Algebraic codebook with 17 bits is used	Algebraic codebook with 20 bits is used
Vector quantization of gains with backward prediction using conjugate structured codebook is used	Ordinary vector quantization of gains with backward prediction is used
Parity bit is used	Parity bit is not used
Frame erasure concealment is done	Frame erasure concealment is not needed

Table 5.1: Comparison of Standard G.729 and the Proposed coder

5.2.1 Implementation Issues

The proposed coder is suitable for multimedia applications. One of the objectives of this work was to evaluate the coder for future real time implementation over PCs. It was found that the algorithm needed much more time in encoding the speech parameters than the availability of speech for transmission. For example. this technique operates on blocks of frame of length 10ms each and computes the speech parameters that are to be transmitted. For a real time application for communication over two PCs we need that the processing time for each frame be less than 10ms. However, with the available resources this could not be met and we had to assess the coder quality by processing the speech off-line. The computational complexity of the coder can be guaged by computing the number of instructions required per second. This is shown in Table 5.2. It can be seen that the proposed coder requires slightly

Standard G. 729	Proposed Coder
24MIPS	27MIPS

Table 5.2: Complexity of the coder in terms of MIPS

larger number of instructions compared to the standard coder. however it can still be easily implemented over dedicated DSP-chips. This increase in computational complexity is because we have more number of samples for analysis for each frame and also the algebraic codebook has a larger size, thereby proportionally increasing the computations.

As other applications such as storage of speech data on discs etc., do not have the constraint of real time encoding, this coder can be efficiently used for storage applications. For a speech sampled at 11ks/s and represented using 16 bits this coder provides a compression ratio of 1:20. Recommendation for communication application of this coder is made in chapter 6.

5.2.2 Quality Assessment

To establish a fair means of comparing speech coding, a variety of quality assessment techniques have been formulated. Generally speaking, tests fall into two classes: subjective quality measures and objective quality measures. Subjective measures are based on comparisons of original and processed speech data by a listener or group of listeners, who subjectively rank the quality of speech along a predetermined scale. Objective quality measures are based on a mathematical comparison of the original and processed speech signals. Most objective quality measures quantify quality with a numerical distance measure or a model of how the auditory system interprets quality. Since the distortion introduced by speech coding systems varies, a collective body of quality measures and tests has emerged for different applications. While objective quality measures are used extensively for speech enhancement algorithms and background noise estimations, the subjective quality tests are more suitable for applications such as coding.

The most important subjective test used for measuring the speech quality is the

Mean Opinion Score (MOS).

5.2.3 Mean Opinion Score

This is an opinion rating method that can be used to assess the degree of quality for a speech processing system. In this method, listeners rate the speech under the test on a five-point scale where a listener's subjective impressions are assigned a numerical value (Table. 5.3). An advantage of the MOS test is that listeners are free to assign their own meanings of "good" to the processed speech. This makes the test applicable to a wide variety of distortions. At the same time, however, this freedom offers a disadvantage in that a listener's scale of "goodness" can vary greatly.

Rating	Speech Quality	Level of Distortion
5	Excellent	Imperceptible
4	Good	Just perceptible but not annoying
3	Fair	Perceptible and slightly annoying
2	Poor	Annoying but not objectionable
1	Unsatisfactory	Very annoying and objectionable

Table 5.3: MOS assessment of speech coders.

The MOS test was carried out for the two coders Standard G.729 and the proposed coder. Standard G.729 was implemented by developing VQ codebooks for gain and LSF and supplemented by the information available through the recent papers on this coder. Proposed coder was implemented as per the information given in chapter 3. The test was carried out by taking speech in two different languages Arabic and English. While the Arabic utterance was that of male speaker, English utterance was that of a female speaker. 14 listeners were used for conducting this test. Listeners were both Arabic and English speaking. The results from the test are tabulated in Table 5.4. It can be seen that the proposed coder performed fairly

Coder	MOS
Standard G.729	4.0
Proposed Coder	4.3

Table 5.4: MOS of coders under test.

well compared to the standard G.729. In fact it fares better than the standard G.729 in terms of quality, because a wider range of frequencies are present in the proposed coder due to its higher sampling rate.

5.2.4 Reproducibility

Another means of estimating the quality of the coder is the reproducibility of the speech waveform. What may appear on the plot may not be too good in perceptible sense, however, it gives a sufficiently good objective measure of testing the coded speech. The plot of a speech utterance in arabic for 5 sec is shown in Figs. 5.1 and 5.2. While Fig. 5.1 represents the reproduced speech waveform in case of the standard G.729 the same is plotted using the proposed coder in Fig. 5.2. Again it can be seen that the proposed coder reproduces the speech waveform fairly well.

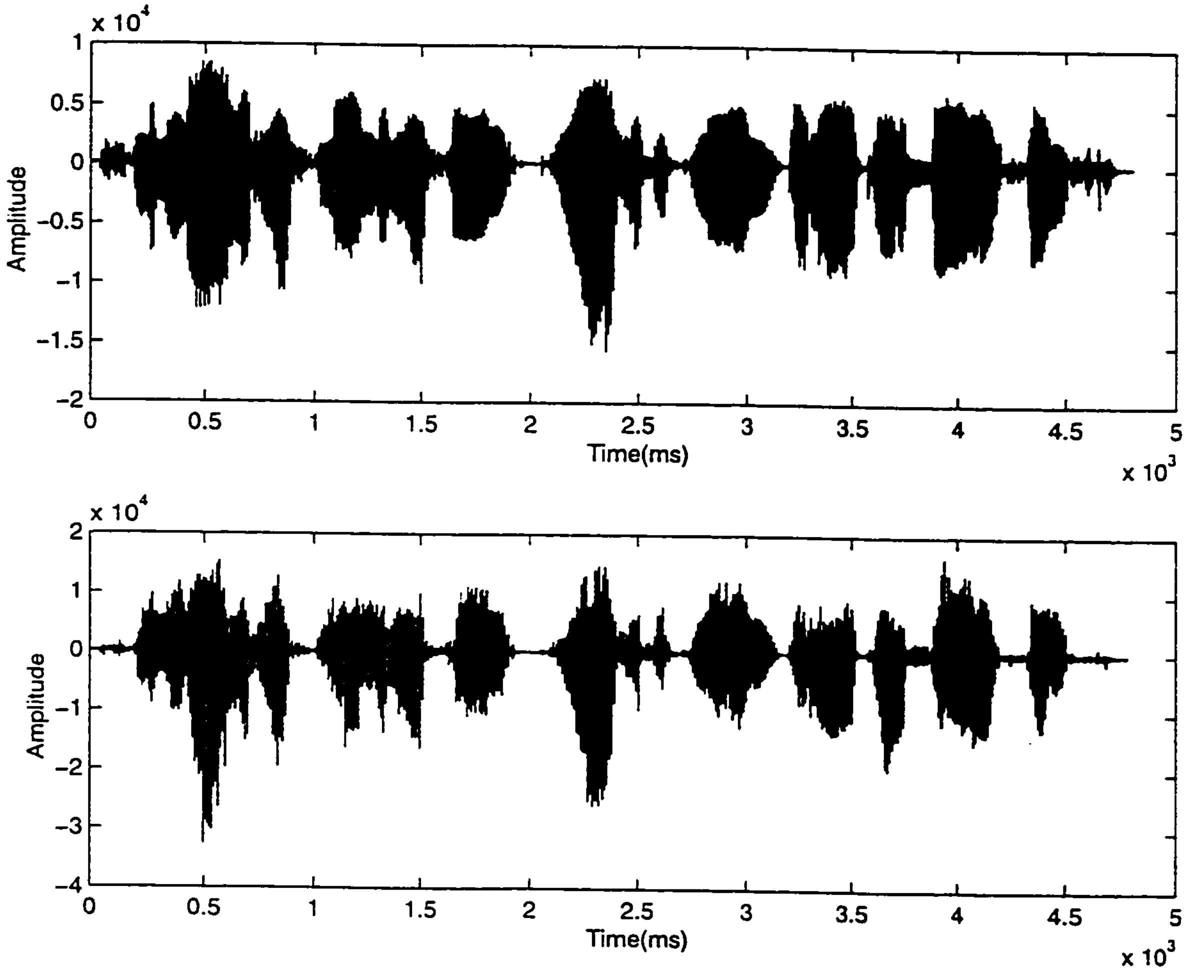


Figure 5.1: Original and Reproduced waveform for G.729

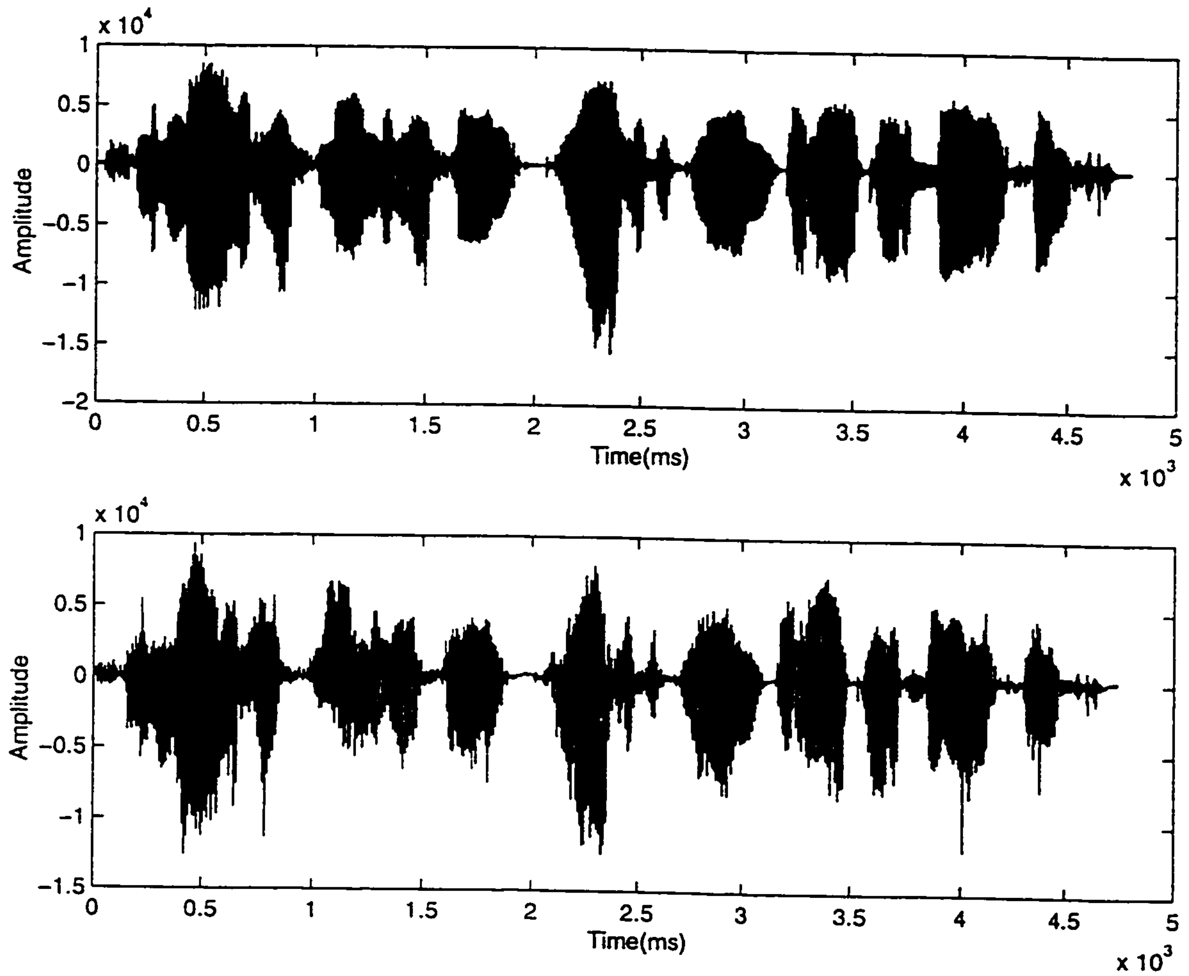


Figure 5.2: Original and Reproduced waveform for Proposed coder

5.3 Validation of Quantization Schemes

The quantizer performance is evaluated by the spectral distortion (SD) which, in dB, is defined as

$$\mathrm{SD} = \sqrt{\frac{1}{B}\int_R [10\,\log(|A_q(e^{j2\pi f})|^2/|A(e^{j2\pi f})|^2)]^2\, df} \tag{5.1}$$

where A and A_q refer to the filter resulting from the original and quantized parameters, respectively. The region R represents the frequency band and B is the bandwidth represented by R.

The LPC power spectrum $S(z) = 1/A(z)$ can be decomposed by

$$|S(e^{-j\omega})|^2 = 4/(|P(e^{-j\omega})|^2 + |Q(e^{-j\omega})|^2) \tag{5.2}$$

where P and Q are the polynomials defined by Eq. 4.4. They are found to be [43]

$$|P(e^{-j\omega})|^2 = (2 + 2cos(\omega))\Pi_{k=1,(odd)}^{M-1}(2 - 2cos(\omega - \omega_k))(2 - 2cos(\omega + \omega_k))$$
$$|Q(e^{-j\omega})|^2 = (2 - 2cos(\omega))\Pi_{k=2,(even)}^{M}(2 - 2cos(\omega - \omega_k))(2 - 2cos(\omega + \omega_k)) \tag{5.3}$$

Eqs. 5.2 and 5.3 are used to compute the spectral distortion by substituting in Eq. 5.1

5.3.1 Two-Stage Split Vector Quantization using Interframe Prediction

In this section the spectral distortion results of the standard G.729 are shown. Predictive split two-stage quantization scheme of G.729 was discussed in detail at the end of chapter 2. It uses a 7 bit first stage codebook, 5 bits each for two second stage split codebooks and 1 bit for the moving average coefficients corresponding to 18bits per frame. The input speech is sampled at 8KHz, the region R lies from 140 Hz to 3400 Hz and B is the bandwidth represented by R. A tenth order LPC analysis is done over each speech frame. These LPC coefficients are then converted to Line Spectral Frequencies. Validation is done using 4000 frames of speech corresponding to 4000 samples of LSFs. The average spectral distortion was found to be around 2.5574 dB. This is shown in Fig. 5.3. It is to be noted that only 18 bits are used in this quantization scheme where as the other two schemes use 21 bits for quantization.

5.3.2 Split Vector Quantization

This scheme for vector quantization of the LSFs comes as a preceding step to the predictive split vector quantization used in the proposed coder. In this technique the 12th order LSF vector is split into three vectors each of dimension 4. Each of these groups of vectors is vector quantized using the LBG algorithm using 7 bits i.e., 128 levels each. So the total number of bits used in this quantizer are 21 per

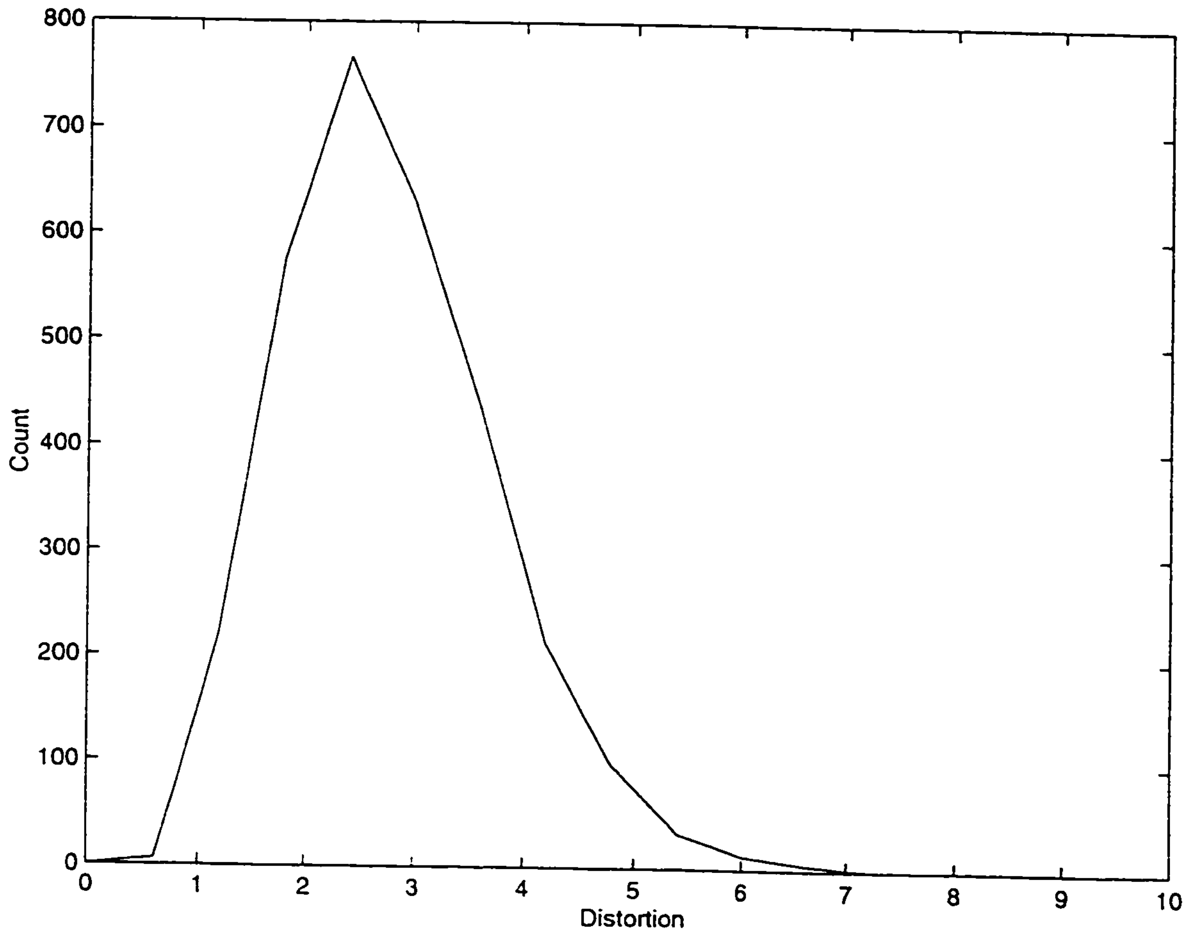


Figure 5.3: Spectral distortion for G.729 LSF quantization scheme

frame. The input speech is sampled at 11ks/s, R lies between 20 to 5400 Hz and B is the bandwidth represented by this R. Validation is done using 1000 frames of speech corresponding to 1000 LSFs. The spectral distortion in dB for this scheme is shown in Fig. 5.4. The average spectral distortion was found to be 2.6094 dB.

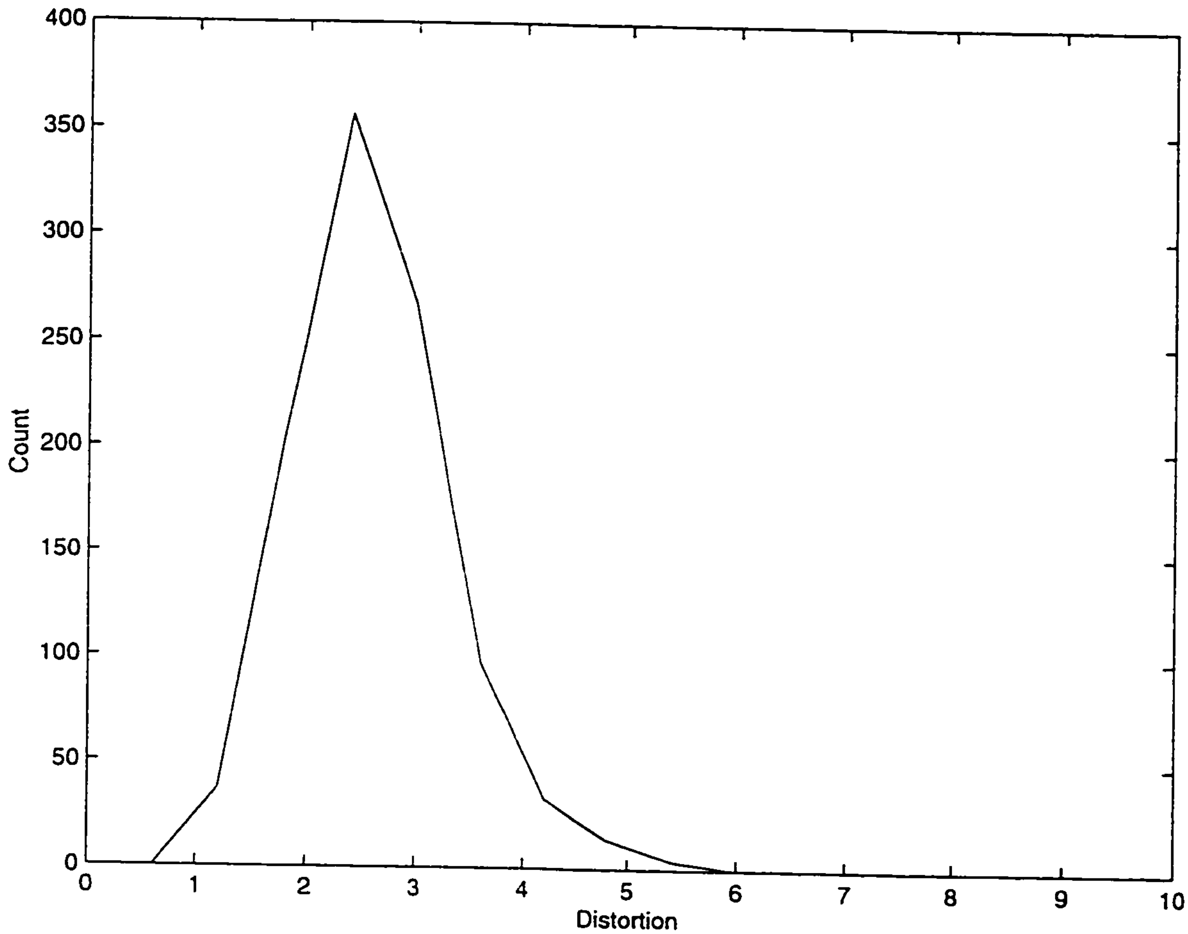


Figure 5.4: Spectral distortion for split vector quantization scheme

5.3.3 Vector Quantization of the Proposed Coder

The proposed coder uses predictive split vector quantization. This is just an improvement to the previously mentioned scheme. As the speech frames are highly correlated, information from the LSF parameters from the previous frame is exploited and used in the quantizer design. The quantizer is already discussed in chapter 3. The results from spectral distortion test is shown in the Fig. 5.5. The input speech here is the same as the one needed for our application i.e., sampled at 11ks/s. R lies between 20 to 5400 Hz and B is the bandwidth represented by R. Validation is done using 1000 frames of speech corresponding to 1000 LSFs. The mean spectral distortion was found to be 2.2302 dB. It can be seen that this value is a little lesser than the one used in standard G.729 and this is the reason why this scheme was used in the proposed coder.

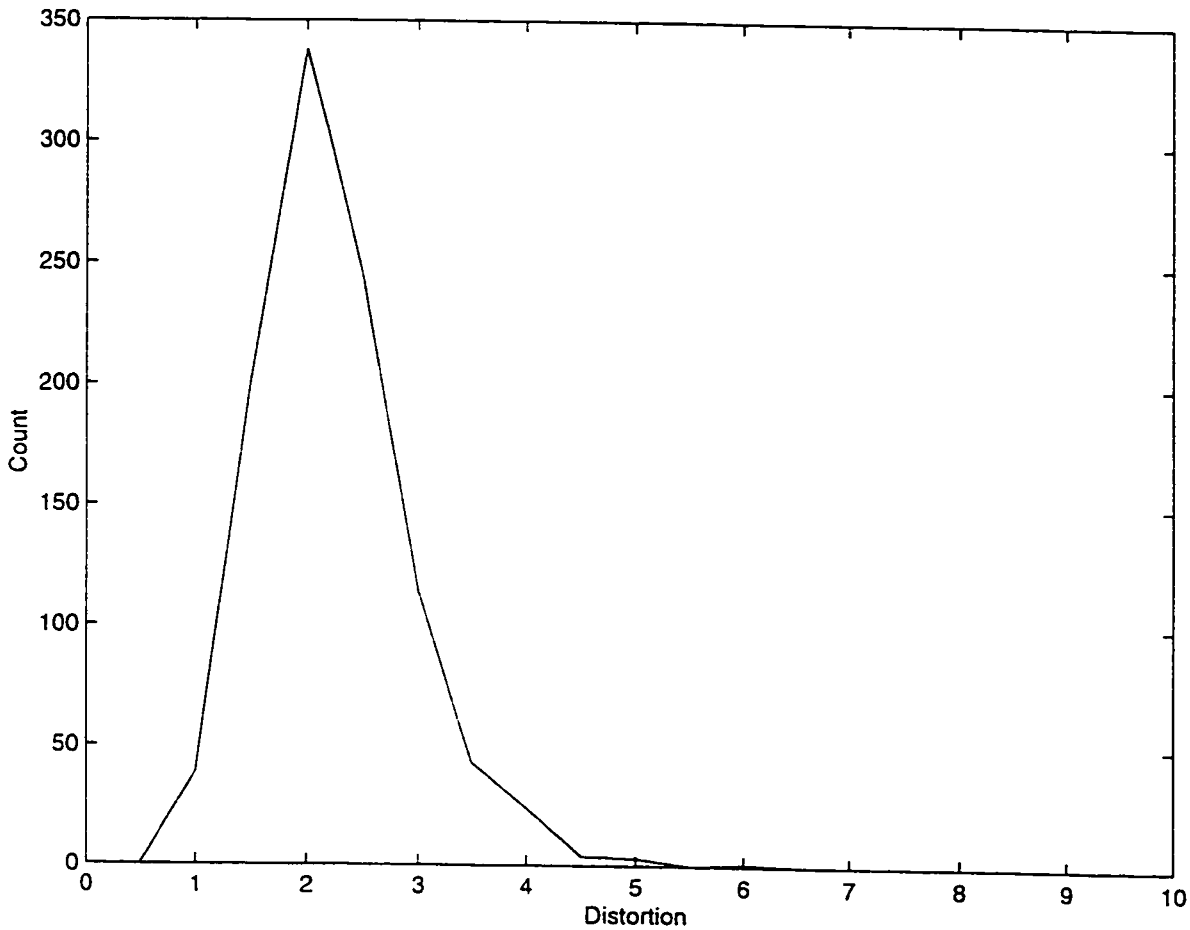


Figure 5.5: Spectral distortion for vector quantization scheme of proposed coder

5.4 Computation of LSF

In this section we discuss the results of the algorithm for LSF computation. Test was done using 10th order LPC filter. The input speech was sampled at 8 Ks/s and passed through a hamming window. The speech frame was selected to be 10 ms of length corresponding to 80samples/frame and the window was of size 30 ms. The LPC coefficients of this hamming windowed speech were computed. The resulting LPC filter coefficients were then converted to the LSF parameters using the algorithm presented in chapter 4. Test was done using speech of 1000 frames and the error in Hz was plotted for the 10 frequencies. It was found that the algorithm failed to compute the exact value in only 4 out of 1000 cases, giving an accuracy of 99.6% which is fairly good enough. The other LSFs computed using this algorithm were well within 4Hz of standard deviation from the actual values, which is acceptable. Computationally the 1st technique converged 38 times faster than the algorithm for accurately computing these frequencies. Plots for different frequencies over the test data is shown in Figs. 5.6 - 5.8.

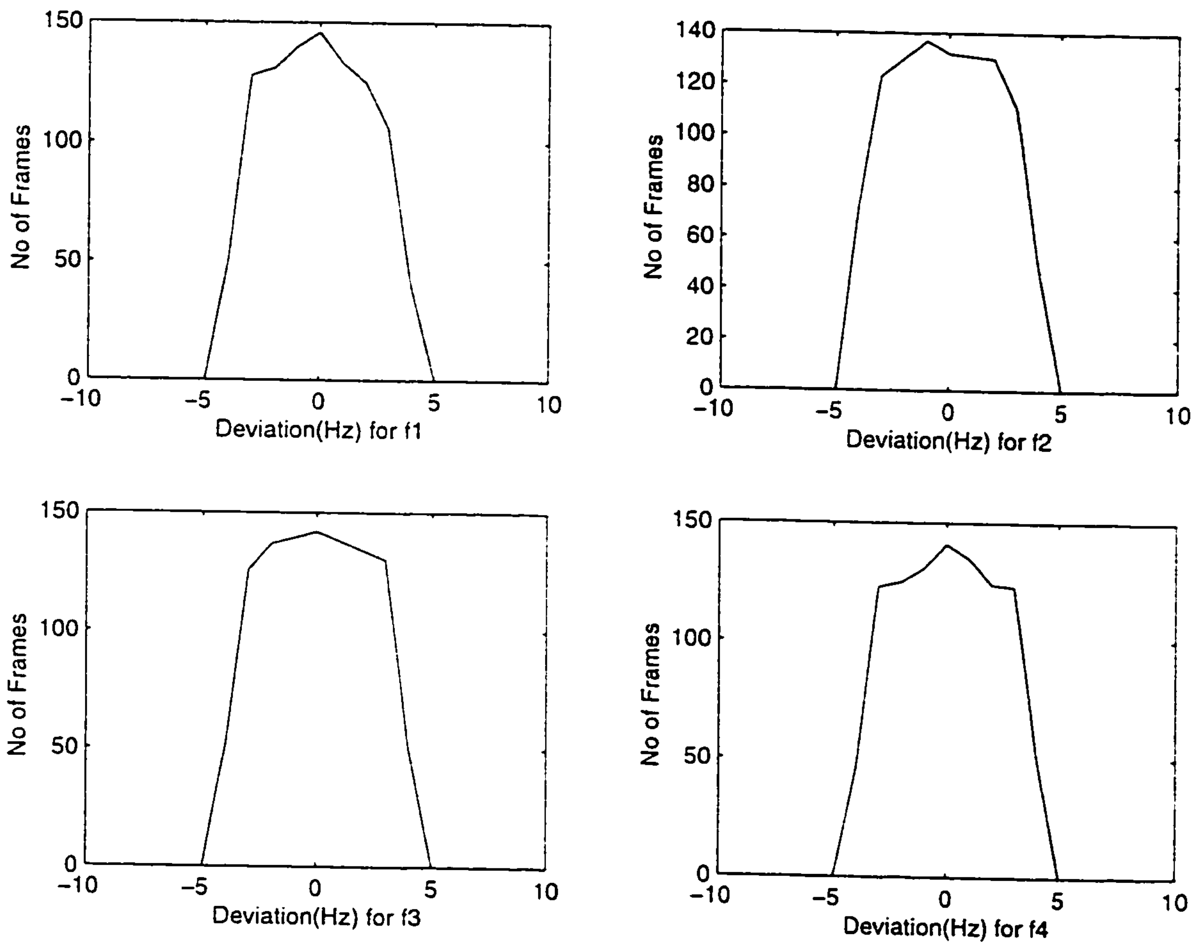


Figure 5.6: Deviation(Hz) for Technique 1

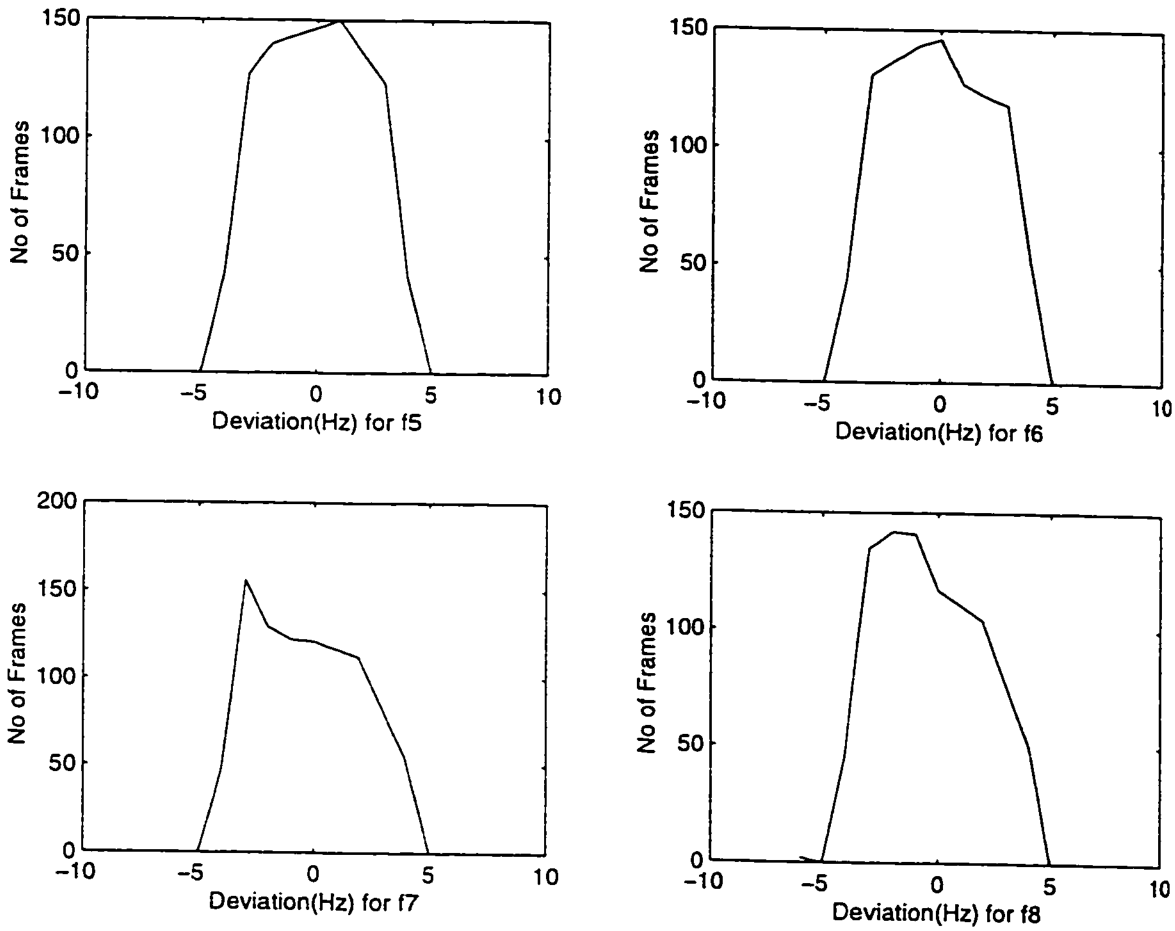


Figure 5.7: Deviation(Hz) for Technique 1

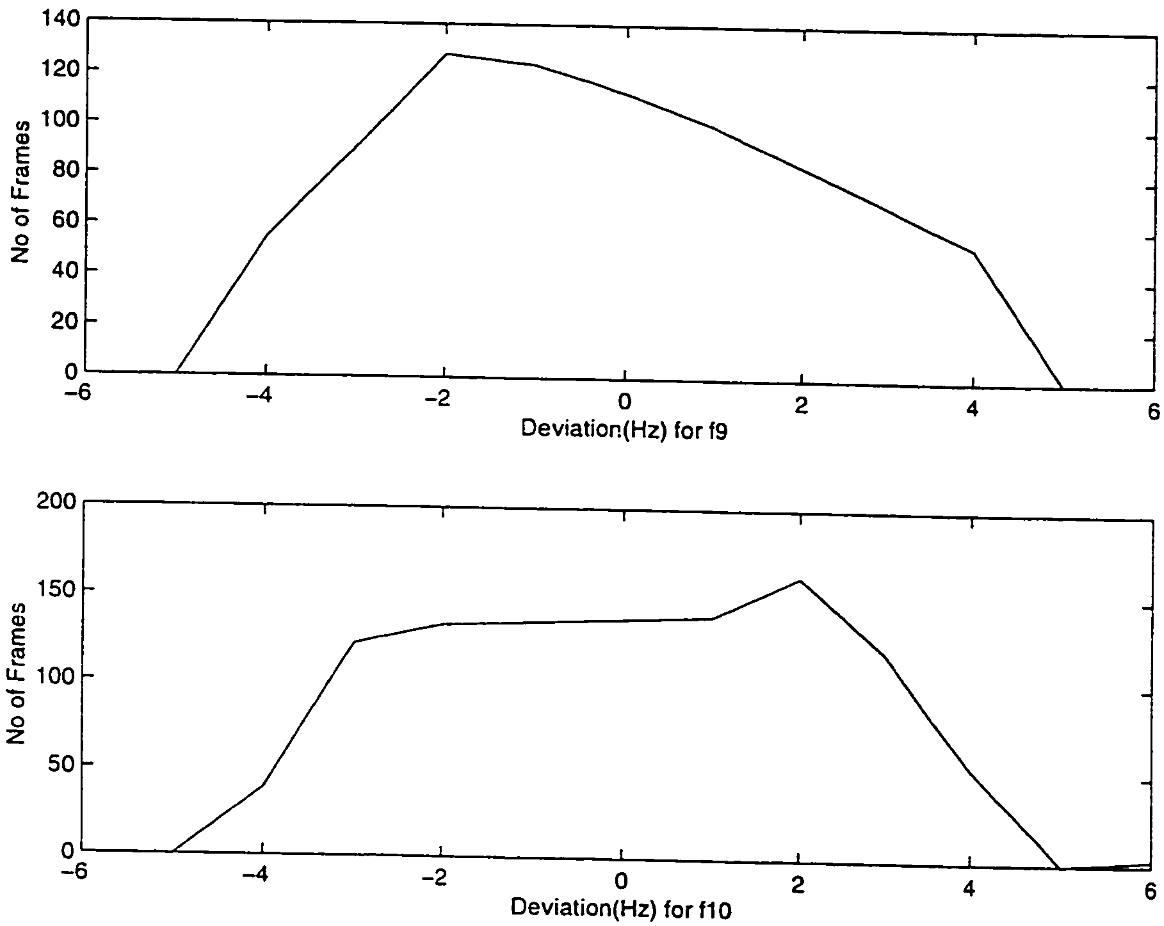


Figure 5.8: Deviation(Hz) for Technique 1

Similarly the second technique was applied using fixed point implementation. The data was the same as the one used with technique 1. It was found that this technique gave the exact result within maximum deviation of 4Hz for all the frequencies. Hence this technique performed even better than the previous one yielding 100% correct results.

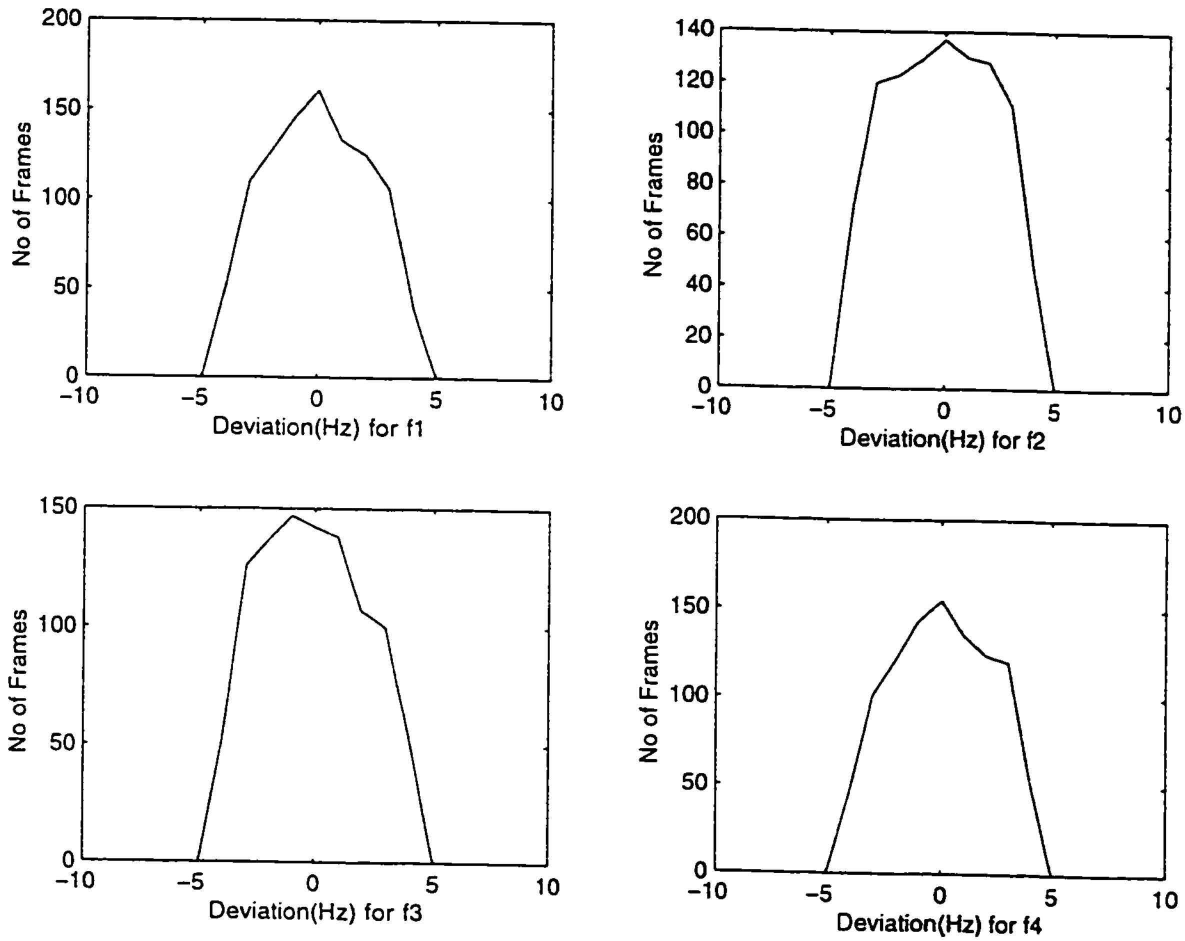


Figure 5.9: Deviation(Hz) for Technique 2

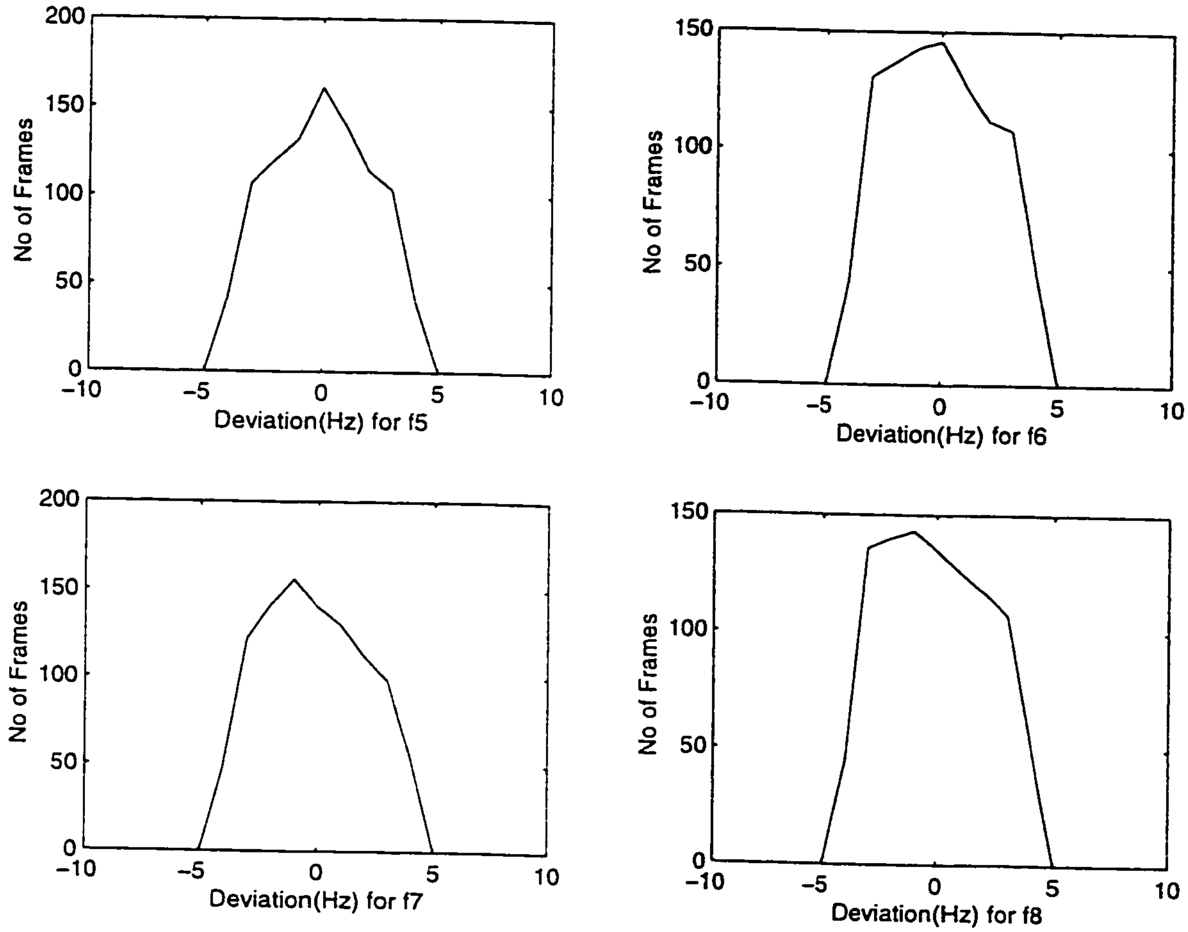


Figure 5.10: Deviation(Hz) for Technique 2

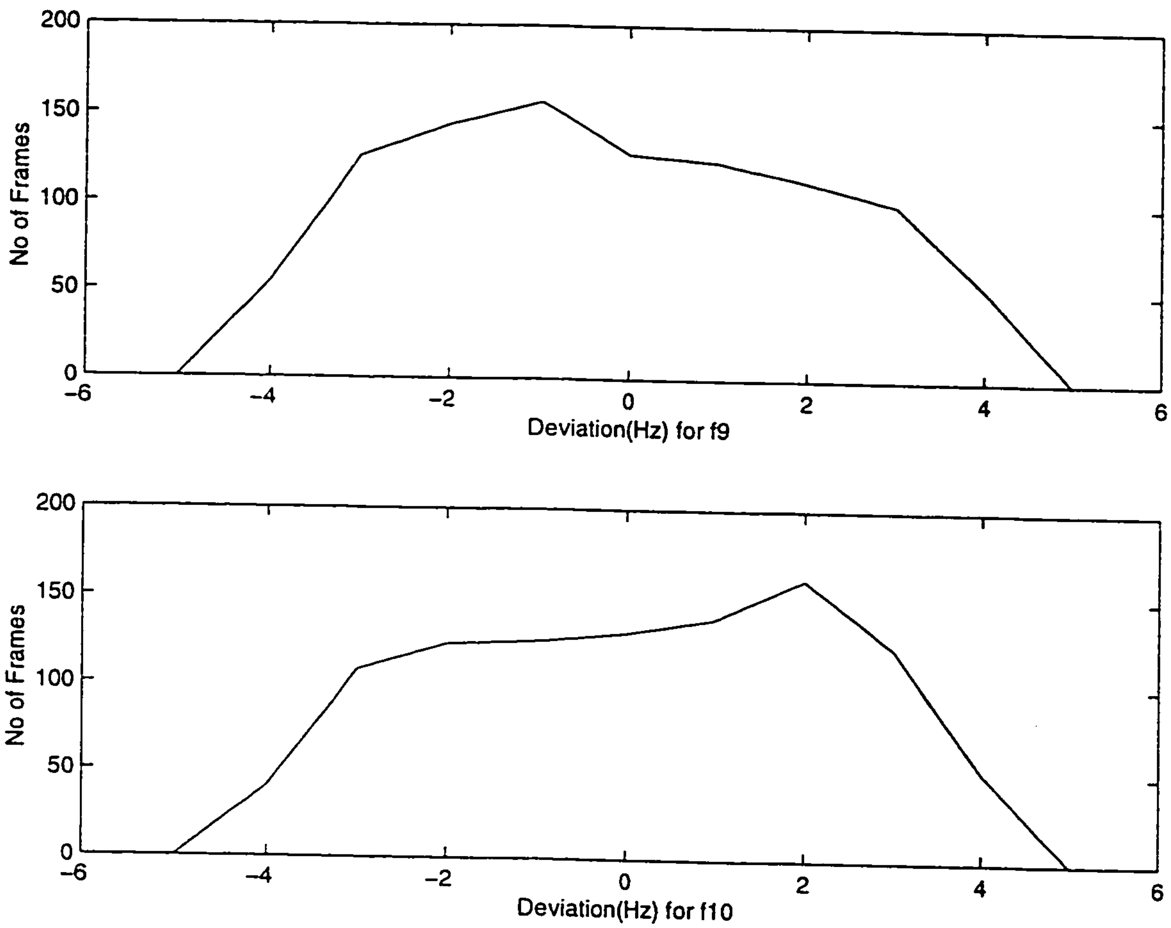


Figure 5.11: Deviation(Hz) for Technique 2

The standard deviation of the frequencies using both the techniques is shown in Table 5.5. It can be seen that the SD of frequencies ω_8 and ω_{10} found using technique. 1 is greater than that using technique. 2 because of the .4% error in computation of these frequencies.

Frequency	Technique. 1	Technique. 2
ω_1	2.2061	2.2061
ω_2	2.2576	2.2576
ω_3	2.2502	2.2502
ω_4	2.2372	2.2372
ω_5	2.1987	2.1987
ω_6	2.2015	2.2015
ω_7	2.2984	2.2984
ω_8	2.5752	2.2448
ω_9	2.3619	2.3620
ω_{10}	4.857	2.2315

Table 5.5: Comparison of Standard Deviation for both the techniques

5.4.1 Computational Complexity

The computational complexity of the proposed LSF computation algorithm using Technique 2 is compared with other available techniques. The required number of operations is shown in Table 5.6. While the other techniques rely on zero crossing search for LSF computation, the proposed algorithm does not require this information. It can be seen that the proposed algorithm gives an excellent reduction in computation compared to some other methods shown in Table 5.6. When compared to the most recently proposed Mixed-LSP algorithm [41], the proposed algorithm provides a more efficient alternative approach.

Technique	No. of A	No. of M	No. of D	No. of S
Chan	3840	3840		
Quantized-search Chan	2040	2040		
Kabal	1390	620	10	
Quantized-search Kabal	642	272		
Mixed-LSP	664	280	12	5
Saoudi	861	706	20	
Proposed Algorithm	216	286	18	

Table 5.6: Computational complexity of different LSF calculation methods using 10th-order LPC system. ($A = add/sub, M = multiplications, D = divisions, S = squareroots.$)

5.5 Implementing the coder over Multimedia Signal Processor

The idea of sharing the same hardware for video and speech signal processing reaches the point that single-chip media processor is required for flexible and low system cost. Multimedia Signal Processor (MSP) has been developed for this reason. This MSP is the programmable SIMD (Single Instruction Multiple Data) processor with eight 32-bit, sixteen 16-bit, or thirty two 8-bit fixed point ALU [42]. The vectorization schemes for the proposed coder's data structure applied to the MSP architecture are discussed here.

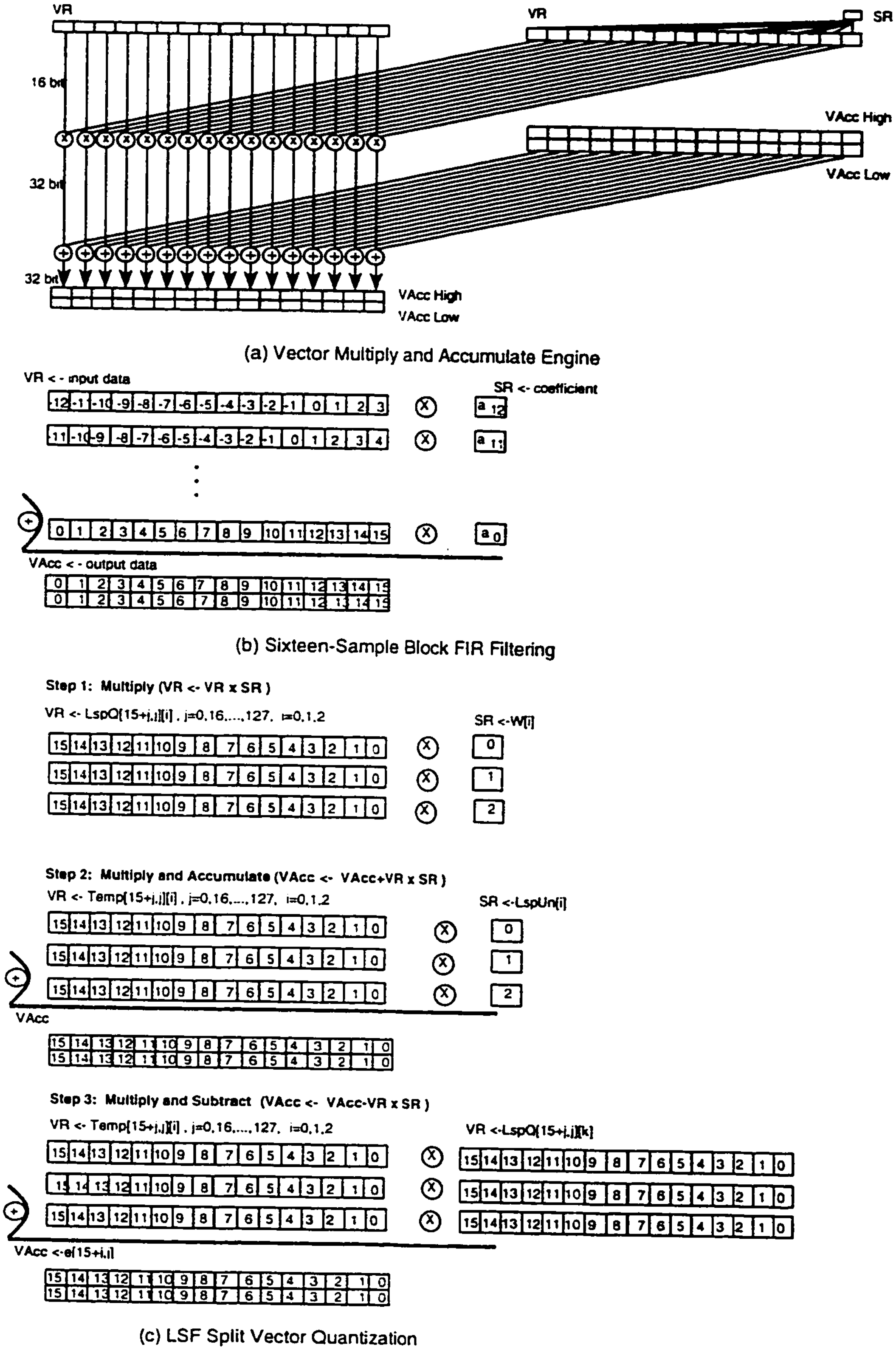


Figure 5.12: Implementation of coder over MSP

Vectorization on SIMD

The key feature of the MSP is its vector multiply and accumulate engine shown in Fig. 5.12 The intermediate product of the two registers (vector * vector or vector * scalar) is added with the accumulator, producing eight 64-bit, sixteen 32-bit or thirty two 16-bit results. When the first operand is a vector register and the second one is a scalar register, the scalar register is multiplied to each element of the first vector register. Some of the vector blocks that are used for implementation of the proposed coder include block FIR filtering, vector quantization (LSF split vector quantization), and algebraic codebook search.

The thirteen coefficients of the FIR filters used in the coder is updated every 55 samples. So these filters are treated as block FIR filter with block size of 55. In MSP implementation, a block size of 16 sample FIR filter is repeated four times to produce 55 outputs. This scheme yields the best performance by utilizing the vector register (VR) which can handle upto 16 samples. Initially, VR gets 12 previous filter memory and 4 input data and is multiplied with the scalar register (SR) which has the thirteenth coefficient. In the next step, the oldest element in VR is shifted out and new input is fed into VR. This VR is multiplied with the twelfth coefficient. Twelve repetition of this step produces sixteen simultaneous outputs in the vector accumulator.

Vector quantization is one of the best algorithm for SIMD processor. In split vector quantization, the twelve dimensional LSF vector transformed from LPC vector is divided into 3 sub-vectors of dimension 4 each. These sub-vectors are vector quantized using the following error measure and the three different VQ tables (128 * 4, 128 * 4, and 128 * 4) for each subvector to give three 7-bit indices.

$$e[j] = \sum_{i=0}^{3} LspQ[j][i] * W[i] * (LspUn[i] - LspQ[j][i]), \quad j = 0, ..., 127 \qquad (5.4)$$

where, $LspQ[j][i]$ is the Lsp quantization table

$W[i]$ is weight vector

$LspUn[i]$ is unquantized Lsp vector.

For parallel operation, $LspQ[j][i]$ entries with different i is loaded into different VR's and $W[k]$ and $LspUn[i]$ are loaded into different SR's. The detailed description of the 3-dimensional case is given in Fig. 5.12(b) and illustrated in the following steps:

1. $Temp[j][i] = LspQ[j][i] * W[i]$

   $VR = MUL\ VR,\ SR$: 3 times

2. $Vacc[j][i] = Vacc[j][i] + Temp[j][i] * LspUn[i]$

   $Vacc = MAC\ VR,\ SR$: 3 times

3. $Vacc[j][i] = Vacc[j][i] - Temp[j][i] * LspQ[j][i]$

   $Vacc = MAS\ VR,\ VR$: 3 times

Now , sixteen $e[j]'s$ are in the $Vacc$. Repetition of steps 1, 2, and 3 eight times gives 128 $e[j]'s$. Thus calculated 128 $e[j]'s$ are stored in eight VR's and the index of the maximum value is searched. Thus VQ of the LSF is done.

The method of finding the index of the maximum $e[j]$ is straight forward. 128 values in eight VR's are compared with each other and the maximum sixteen values from each column are saved in one VR (of dimension eight). Then maximum value is searched among the eight values in this VR and compared with the original 128 values in eight VR's. When the maximum value matches with one of the original values in VR's the corresponding bits of the destination SR are set to 1's. The index of the maximum value is calculated by counting the leading zeros in the SR.

Similar analysis is done for the gain vector quantization. In case of the Algebraic codebook the analysis is done in four nested loops. To reduce the computation, the inner-most loop is unrolled by copying the mirror image of the correlations of impulse response.

Chapter 6

CONCLUSION AND RECOMMENDATIONS

6.1 Conclusion

Waveform encoding of speech has been widely used for several decades on telephone channels. However, for speech communication over mobile cellular radio channels and multimedia communications, high quality speech at low bit rate is required. In such applications LPC-based encoding schemes employing VQ, such as CELP is found to be particularly suitable. In this thesis work a high-quality speech coder based on CELP analysis by synthesis technique was proposed. This coder at a sampling rate of 11ks/s is suitable for PC based multimedia applications such as storage over high density disks or communication via a modem. The ternary-level algebraic codebook improves the speech quality greatly, at the same time the efficient search method and a simple predictive split-vector quantization of the LSFs reduces the computational complexity. Non integer pitch delay improves the speech quality. Speech coding is a compromise between quality and complexity. Further quality is not a definite parameter: it varies from person to person based on his perception. An effort was made to come up with a coder that is less complex in structure but at the same time offers sufficiently good quality of speech.

Though the LSFs give a good alternate representation of the LPCs their computation consumed a great amount of time(operations) during speech coding. An algorithm for efficient computation of the LSFs was also proposed. It was found to produce results i.e.. the LSFs within the required error limits with considerably less computation time.

6.2 Recommendations

First and foremost recommendation is towards the real time application of the coder, such as communication between the PCs via modems. The algorithm that we had written was implemented over a windows based platform, which itself had a huge computational overhead. There is a need to implement this algorithm as part of a DSP chip such as TMS320C30 or TMS320C50, which will facilitate real time applications. It was found that this algorithm requires almost the same number of computations as the standard G.729 and hence can be easily implemented over a DSP chip.

Our coder operates at a bit rate of 8.8kb/s. Few questions that arise are, should this bit rate be fixed? What if we have a higher or lower channel capacity to which we have to adapt this coder? What changes have to be made to either increase or decrease this bit rate? The answer to these questions lies in the fact that, speech quality as produced by a speech coder is a function of bit rate, complexity, delay, and bandwidth.

First let us consider the choice of increasing the bit rate. The most dominant factor that controls the quality of speech is the quantization of the LSFs. Priority is to be given to these parameters so as to obtain transparent LSFs. These extra bits could be accommodated by increasing the number of quantizer levels. Next priority would be to increase the number of levels in vector quantizer of the gains. Finally, the pitch resolution can also be increased from $\frac{1}{2}$ to $\frac{1}{3}$ or more. Now suppose that the bit rate was to be decreased, the choice now is very difficult and a compromise will have to be made among the three factors i.e., LSF quantization, gain quantization and pitch resolution to come up with a best solution in terms of quality.

Appendix A

Computational Aspects for Technique 1

- The coefficients p_i and q_i for a 10th order LPC filter are given as:

$$p(i+1) = a_{i+1} + a_{10-i} - p(i) \qquad \text{i=0,....,4}$$
$$q(i+1) = a_{i+1} - a_{10-i} + q(i) \qquad \text{i=0,...,4} \tag{A.1}$$

- The Newton's formula used in these computations is

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} \tag{A.2}$$

- The equations outlined here are the same for either of the two polynomials P or Q. We will show here the computations for P, which are equally applicable for Q. In general we will treat the polynomial as Z, with roots $\alpha_i, \quad i = 1, ..., N$. Further all the computations shown here are for the 10th order LPC polynomial i.e., $N = 5$.

- The polynomial $P_c(\omega)$ here in after referred to as Z_n, with its subscript n representing the order, can be written in terms of a nth order polynomial with $(x = cos(\omega))$ as :

$$Z_5 = 16p(0)x^5 + 8p(1)x^4 + [-20p(0) + 4p(2)]x^3 + [-8p(1) + 2p(3)]x^2$$
$$+[5p(0) - 3p(2) + p(4)]x + p(1) - p(3) + .5p(5) \qquad (A.3)$$

For simplification of analysis the above equation can be equivalently written with the subscript on the coefficient corresponding to the order of the polynomial, as:

$$Z_5 = x^5 + z_5(1)x^4 + z_5(2)x^3 + z_5(3)x^2 + z_5(4)x + z_5(5) \qquad (A.4)$$

where $z_5(1) = \frac{8p(1)}{16p(0)}$, $z_5(2) = \frac{(-20p(0)+4p(2))}{16p(0)}$ and so on.

After getting the first root i.e., α_1, this root is factored out of the fifth order polynomial, the resulting fourth order polynomial is given by:

$$Z_4 = x^4 + z_4(1)x^3 + z_4(2)x^2 + z_4(3)x + z_4(4) \tag{A.5}$$

where $z_4(4) = \frac{-z_5(5)}{\alpha_1}$ and $z_4(j) = \frac{z_4(j+1)-z_5(j)}{\alpha_1}$ for $j = 1, ..., 4$.

Similarly the nth order polynomial and the coefficients after factorizing out the kth root is given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)x^{n-i} \tag{A.6}$$

where, $z_n(0) = 1$, $z_n(n) = \frac{-z_{n+1}(n+1)}{\alpha_k}$ and $z_n(j) = \frac{z_n(j+1)-z_{n+1}(j)}{\alpha_k}$ for $j = n-1, ..., 1$ and $n = N-1, ..., 1$.

- In addition we also need the values of coefficients of differential of the polynomial Z_n which is represented by Z'_n. The differential for the fifth order polynomial is given by

$$Z'_5 = z'_5(1)x^4 + z'_5(2)x^3 + z'_5(3)x^2 + z'_5(4)x + z'_5(5) \tag{A.7}$$

where $z'_5(1) = 5$, $z'_5(2) = 4z_5(1)$ and so on.

In general the nth order differential polynomial can be written as

$$Z'_n = \sum_{i=1}^{n} z'_n(i)x^{n-i} \tag{A.8}$$

where $z'_n(i) = (n-i+1)z_n(i-1)$, for $i = 1, ..., n$ and $z_n(0) = 1$.

Appendix B

Computational Aspects for Technique 2

- Chebyshev polynomials that are used in the computation have the following property :

$$T_{i+1} = \frac{1}{2}T_i * T_1 - T_{i-1} \tag{B.1}$$

where $T_i = cos(i * \omega)$.

- The equations outlined hereafter are the same for either of the two polynomials P or Q. We will show here the computations for P, which are equally applicable for Q. In general we will treat the polynomial as Z, with roots α_i, $i = 1, ..., N$. Further all the computations shown here are for the 10th order LPC polynomial i.e., $N = 5$.

- The polynomial $P_c(\omega)$ here in after referred to as Z_n, with its subscript n representing the order. can be written in terms of the Chebyshev polynomial as:

$$Z_5 = z_5(0) + z_5(1)T_1 + z_5(2)T_2 + z_5(3)T_3 + z_5(4)T_4 + z_5(5) \qquad \text{(B.2)}$$

where the coefficients $z_5(0)$, $z_5(1)$ and so on are obtained by properly scaling the coefficients p(0), p(1), and so on in term 3.5.

After getting the first root i.e., α_1, this root is factored out of the fifth order polynomial. the resulting fourth order polynomial is given by:

$$Z_4 = z_4(0) + z_4(1)T_1 + z_4(2)T_2 + z_4(3)T_3 + T_4 \qquad \text{(B.3)}$$

So given the fifth order polynomial. and the root α_1 all we need to find are the coefficients $z_4(0)...z_4(3)$, to get the fourth order polynomial. These set of coefficients can be obtained from the following set of equations

$$z_4(3) - 2cos(\alpha_1) = z_5(4)$$

$$z_4(2) - 2cos(\alpha_1)z_4(3) + 1 = z_5(3)$$

$$z_4(1) - 2cos(\alpha_1)z_4(2) + z_4(3) = z_5(2)$$

$$2 * z_4(0) - 2cos(\alpha_1)z_4(1) + z_4(2) = z_5(1)$$

$$-2cos(\alpha_1)z_4(0) + z_4(1) = z_5(0) \tag{B.4}$$

$$\tag{B.5}$$

Similarly the nth order polynomial and the coefficients after factorizing out the kth root is given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)T_i \tag{B.6}$$

where,

$$\beta * z_n(l-1) = z_{n+1}(l) + 2z_n(l)cos(\alpha_k) - z_n(l+1) \quad \text{for l=N-k,....1.}$$

and

$$for \;\; l = 1, \beta = 2; \;\; else \;\; \beta = 1$$

$$z_{n+1}(N-k) = 1$$

$$z_n(N-k) = 0$$

$$z_n(N-k-1) = 1$$

Appendix C

Computational Aspects for Fixed Point Implementation

This section deals with issues related to fixed point implementation of Technique 2. Issues related to normalizing the coefficients are shown. Analysis is based on a 10th order LPC filter.

Normalizing the coefficients

Recollect that the fifth and the fourth order polynomial coefficients of Eq. B.2 is related by

$$z_4(3) = z_5(4) + 2cos(\alpha_1)z_4(4) - z_4(5)$$

$$z_4(2) = z_5(3) + 2cos(\alpha_1)z_4(3) - z_4(4)$$

$$z_4(1) = z_5(2) + 2cos(\alpha_1)z_4(2) - z_4(3)$$

$$z_4(0) = z_5(1) + 2cos(\alpha_1)z_4(1) - z_4(2)$$

$$\text{and a correction } z_4(0) = z_4(0)/2. \tag{C.1}$$

where $z_4(5) = 0$ and $z_4(4) = 1$.

The equations given above are sufficient to compute the coefficients on th LHS. Similarly the nth order polynomial and the coefficients after factorizing out the kth root are given by:

$$Z_n = \sum_{i=0}^{n} z_n(i)T_i \tag{C.2}$$

where,

$$z_n(l-1) = z_{n+1}(l) + 2z_n(l)cos(\alpha_k) - z_n(l+1) \quad \text{for l=N-k,....,1.}$$

and

$$z_n(0) = z_n(0)/2$$

$$z_{n+1}(N-k) = 1$$

$$z_n(N-k) = 0$$

$$z_n(N-k-1) = 1$$

We know that the value of ω lies between $0 < \omega < \pi$. we now define a new parameter f such that $f = \omega/\pi$. so the value of f lies in the boundaries $0 < f < 1$.

For the 10th order LSF let us begin with the coefficients of Z_5 given by $z_5(i)$ $i = 0, ..., 4$. If we normalize these coefficients as $z_5'(i)$ then their values are between 0 and 1. Thereafter the fourth order polynomial would have the coefficients

$$z_4(3) = z_5'(4) + 2cos(\alpha_1)z_4(4) - z_4(5)$$

$$z_4(2) = z_5'(3) + 2cos(\alpha_1)z_4(3) - z_4(4)$$

$$z_4(1) = z_5'(2) + 2cos(\alpha_1)z_4(2) - z_4(3)$$

$$z_4(0) = z_5'(1) + 2cos(\alpha_1)z_4(1) - z_4(2)$$

$$\text{and a correction } z_4(0) = z_4(0)/2. \tag{C.3}$$

Now if we define normalized fourth order coefficients as $z_4'(i)$ where $z_4'(i) = z_4(i)/4$ then the normalized coefficients are given by

$$z_4'(3) = z_5'(4) + 2cos(\alpha_1)z_4'(4) - z_4'(5)$$

$$z_4'(2) = z_5'(3) + 2cos(\alpha_1)z_4'(3) - z_4'(4)$$

$$z_4'(1) = z_5'(2) + 2cos(\alpha_1)z_4'(2) - z_4'(3)$$

$$z_4'(0) = z_5'(1) + 2cos(\alpha_1)z_4'(1) - z_4'(2)$$

$$\text{and a correction } z_4'(0) = z_4'(0)/2. \tag{C.4}$$

The above set of coefficients are always between 0 and 1 provided $z_5'(i) = z_5'(i)/4$, which was obtained earlier. So a general routine to always get normalized coefficients between 0 and 1 is outlined here.

1. The first set of polynomials i.e $z_n(i),\ i = 0, ..., n-1$ are normalized to obtain the values between 0 and 1 and denoted by z_n'.

2. Thereafter to obtain the normalized coefficients of any order n the following iteration has to be used with $z_{n+1}' = z_{n+1}'/4$ to obtain $0 \leq z_n' \leq 1$.

$$z_n'(l-1) = z_{n+1}'(l) + 2z_n'(l)cos(\alpha_k) - z_n'(l+1) \quad \text{for l=N-k,...,1.}$$

and

$$z_n'(0) = z_n'(0)/2$$

$$z_{n+1}'(N-k) = 1$$

$$z_n'(N-k) = 0$$

$$z_n'(N-k-1) = 1$$

Appendix D

Computing the Pitch in Closed Loop

If g_p is the pitch gain and if no stochastic excitation is present, the excitation due to a pitch delay of k is given by

$$u(n) = g_p u(n-k) \tag{D.1}$$

The weighted synthesized speech is given by

$$\hat{s}_w(n) = \sum_{i=0}^{n} u(i)h(n-i) + \hat{s}_0(n), \tag{D.2}$$

where n is the number od samples over which the test for pitch has to be done. $h(n)$ is the impulse response of the weighted synthesis filter (see section 3.4.7) and $\hat{s}_0(n)$ is the zero-input response of the weighted synthesis filter, that is, the output of the filter due to its initial states. The weighted error between the original and the synthesized speech is given by

$$\begin{aligned} e_w(n) &= s_w(n) - \hat{s}_w(n) \\ &= t(n) - \sum_{i=0}^{n} u(i)h(n-i) \end{aligned} \tag{D.3}$$

where

$$t(n) = s_w(n) - \hat{s}_0(n) \tag{D.4}$$

and $s_w(n)$ is the weighted input speech. Substituting Eq. D.1 into Eq. D.3 gives

$$e_w(n) = x(n) - g_p y_k(n), \tag{D.5}$$

where

$$e_k(n) = u(n-k) * h(n). \tag{D.6}$$

The mean squared weighted error is given by

$$E_w = \sum_{n=0}^{N-1} [t(n) - g_p y_k(n)]^2 \tag{D.7}$$

where N is the subframe length. Setting $\partial E_w/\partial G = 0$ leads to

$$g_p = \frac{\sum_{n=0}^{N-1} t(n) y_k(n)}{\sum_{n=0}^{N-1} [y_k(n)]^2} \tag{D.8}$$

Substituting Eq. D.8 in Eq. D.7 gives

$$E_w = \sum_{n=0}^{N-1} [t(n)]^2 - \frac{[\sum_{n=0}^{N-1} t(n) y_k(n)]^2}{\sum_{n=0}^{N-1} [y_k(n)]^2} \tag{D.9}$$

The pitch delay k is selected as the delay which maximizes the second term in Eq. D.9 and this term is defined as

$$\tau_k = \frac{|\sum_{n=0}^{N-1} t(n) y_k(n)|}{\sqrt{\sum_{n=0}^{N-1} y_k(n) y_k(n)}} \tag{D.10}$$

which is the one defined as Eq. 3.24 in section 3.4.7.

Appendix E

LSF Vector Quantizer

In this section the vector quantizer that is used in the proposed coder is tabulated. The codebook consists of the entries corresponding to the error vectors $\omega_{e1}, \omega_{e2}, \ldots, \omega_{e12}$. Decoding procedure shown in chapter 3 is to be done and the ω_{dc} that is required is shown in the next table.

$\omega_{dc,1}$	$\omega_{dc,2}$	$\omega_{dc,3}$	$\omega_{dc,4}$	$\omega_{dc,5}$	$\omega_{dc,6}$
0.0507	0.1356	0.2046	0.4149	0.6503	0.8668
$\omega_{dc,7}$	$\omega_{dc,8}$	$\omega_{dc,9}$	$\omega_{dc,10}$	$\omega_{dc,11}$	$\omega_{dc,12}$
1.0646	1.2982	1.5809	1.8339	2.1183	2.5007

Table E.1: DC components of the LSF vector

V.No	ωε1	ωε2	ωε3	ωε4	ωε5	ωε6	ωε7	ωε8	ωε9	ωε10	ωε11	ωε12
1	0.1061	0.3240	0.4181	0.3263	0.4342	0.5610	0.5910	0.5625	0.5630	0.4940	0.4414	0.2276
2	0.1913	0.3704	0.4622	0.4139	0.4872	0.4835	0.6177	0.5203	0.4484	0.4925	0.4175	0.3459
3	0.0471	0.4051	0.4673	0.4441	0.0917	0.0575	0.0755	0.1666	0.4523	0.5111	0.4307	0.2549
4	0.1022	0.3458	0.4394	0.4437	0.4637	0.5004	0.5451	0.4966	0.4544	0.5028	0.4255	0.2303
5	0.2204	0.3920	0.3613	0.4151	0.3778	0.4847	0.5687	0.6087	0.4560	0.4428	0.4198	0.3009
6	0.1818	0.3840	0.3847	0.3609	0.2873	0.5094	0.5540	0.5979	0.4699	0.4260	0.3977	0.3127
7	0.0737	0.0639	0.1470	0.1253	0.3630	0.4922	0.5276	0.5430	0.4941	0.3948	0.4240	0.2644
8	0.1738	0.3295	0.4025	0.3134	0.3844	0.4538	0.4705	0.5594	0.4636	0.4461	0.3739	0.2577
9	0.0594	0.3083	0.5127	0.4568	0.4329	0.4201	0.5585	0.5049	0.5460	0.4788	0.3656	0.2135
10	0.0812	0.2911	0.4727	0.4712	0.4507	0.4297	0.5450	0.4522	0.5597	0.4827	0.3510	0.1514
11	0.0257	0.2393	0.5596	0.4291	0.3936	0.5190	0.4961	0.4753	0.5027	0.4311	0.3912	0.2172
12	0.0362	0.2785	0.4760	0.4563	0.4558	0.4126	0.5166	0.4687	0.4895	0.4610	0.3792	0.1882
13	0.1230	0.3100	0.4081	0.4246	0.2103	0.4655	0.6008	0.5603	0.4766	0.4673	0.3517	0.2026
14	0.1117	0.3053	0.4065	0.3975	0.3255	0.3678	0.6184	0.5104	0.5143	0.4402	0.3439	0.1928
15	0.0773	0.3573	0.4062	0.3349	0.3225	0.4223	0.5261	0.4962	0.4849	0.4204	0.3510	0.2060
16	0.0490	0.3277	0.4012	0.3674	0.3182	0.3582	0.5359	0.5477	0.5003	0.3846	0.3458	0.1903
17	0.0890	0.2283	0.4754	0.4474	0.4995	0.4958	0.4855	0.4339	0.3558	0.4529	0.4759	0.2399
18	0.0794	0.2113	0.4427	0.4819	0.4919	0.3500	0.5271	0.4813	0.3828	0.4286	0.3975	0.2840
19	0.0383	0.0389	0.1151	0.2609	0.4494	0.4857	0.4263	0.4584	0.3747	0.3784	0.4385	0.2733
20	0.0726	0.2814	0.3382	0.4958	0.4574	0.4092	0.4603	0.4322	0.3696	0.3661	0.4316	0.2550
21	0.0909	0.2643	0.4116	0.4190	0.3856	0.4545	0.4749	0.4775	0.4280	0.4052	0.3743	0.2684
22	0.0691	0.2542	0.4079	0.4126	0.3866	0.4545	0.4135	0.5057	0.4244	0.3882	0.3547	0.2747
23	0.0940	0.2338	0.3640	0.4454	0.3860	0.4138	0.5278	0.4185	0.3544	0.4433	0.3783	0.2170
24	0.0900	0.2235	0.3446	0.4355	0.3184	0.4297	0.4922	0.4181	0.3607	0.3973	0.3703	0.2254
25	0.0788	0.1696	0.4851	0.4332	0.4926	0.5131	0.4584	0.3563	0.4562	0.3944	0.3521	0.2448
26	0.0453	0.1982	0.4741	0.4122	0.4479	0.4735	0.4166	0.3827	0.4428	0.3936	0.3561	0.2234
27	0.0538	0.1750	0.4693	0.4035	0.5669	0.4721	0.3618	0.3084	0.4257	0.3744	0.3646	0.2152
28	0.0321	0.1551	0.4815	0.3935	0.5141	0.4215	0.3716	0.2593	0.4357	0.3482	0.3446	0.2294
29	0.0671	0.1957	0.3789	0.5041	0.4201	0.3605	0.4739	0.4415	0.4536	0.3958	0.3194	0.1895
30	0.0837	0.1615	0.3715	0.4694	0.3896	0.3893	0.4891	0.3731	0.4371	0.3926	0.2974	0.1958
31	0.0635	0.1843	0.4042	0.4017	0.3932	0.4308	0.3941	0.4392	0.4061	0.3711	0.3385	0.1752
32	0.0285	0.1911	0.3851	0.4085	0.4021	0.3895	0.3770	0.4375	0.4025	0.3897	0.2940	0.1656

Figure E.1: Vector Quantizer

V.No	ωε1	ωε2	ωε3	ωε4	ωε5	ωε6	ωε7	ωε8	ωε9	ωε10	ωε11	ωε12
33	0.1397	0.2931	0.3796	0.3393	0.3384	0.3779	0.4796	0.5236	0.2885	0.4378	0.3982	0.2891
34	0.1396	0.2819	0.3350	0.3619	0.3902	0.3316	0.4169	0.5255	0.2604	0.3959	0.4384	0.2464
35	0.0904	0.2962	0.3897	0.3181	0.2864	0.3437	0.4878	0.5053	0.3666	0.3328	0.3596	0.3093
36	0.0905	0.2648	0.3730	0.3119	0.2596	0.3633	0.4682	0.4807	0.3386	0.3554	0.3490	0.2830
37	0.1301	0.2080	0.3460	0.3907	0.3370	0.3731	0.4310	0.4581	0.3187	0.3042	0.4161	0.2647
38	0.1331	0.1987	0.3293	0.3827	0.3315	0.4180	0.3714	0.4413	0.3151	0.3301	0.3803	0.2543
39	0.1178	0.1789	0.3948	0.3196	0.3272	0.3863	0.4605	0.3611	0.2844	0.3295	0.3856	0.2601
40	0.1183	0.1808	0.3455	0.3387	0.3356	0.3252	0.4092	0.4321	0.2962	0.2741	0.4168	0.2357
41	0.0631	0.2370	0.4010	0.3634	0.4515	0.4282	0.3933	0.3384	0.4062	0.3234	0.3680	0.2226
42	0.0436	0.2384	0.3750	0.3614	0.4424	0.4240	0.3522	0.2877	0.3871	0.3114	0.3889	0.1991
43	0.0324	0.2421	0.4049	0.3067	0.4356	0.3633	0.4044	0.3561	0.3840	0.3362	0.3011	0.2637
44	0.0257	0.2066	0.4112	0.3099	0.4015	0.2987	0.4631	0.3412	0.3614	0.3572	0.3145	0.2068
45	0.0598	0.2242	0.3294	0.3718	0.3962	0.3680	0.3394	0.4044	0.3624	0.2955	0.3884	0.2188
46	0.0545	0.2129	0.3161	0.3646	0.4157	0.3582	0.3326	0.3763	0.3114	0.3744	0.3152	0.2374
47	0.0442	0.2214	0.3506	0.3015	0.3608	0.3838	0.3869	0.2888	0.2891	0.3648	0.3573	0.1890
48	0.0252	0.2065	0.3314	0.2977	0.4003	0.3742	0.2968	0.3241	0.3018	0.3230	0.3706	0.1737
49	0.0359	0.1661	0.3676	0.5113	0.2603	0.2994	0.4694	0.4748	0.4050	0.3482	0.2448	0.2627
50	0.0064	0.1616	0.3368	0.5178	0.3234	0.2892	0.3279	0.5379	0.3708	0.3491	0.2733	0.2225
51	0.0245	0.1134	0.4337	0.4217	0.3219	0.2691	0.3855	0.4803	0.3897	0.2856	0.2843	0.2541
52	0.0312	0.1203	0.4068	0.4078	0.2649	0.2416	0.4584	0.4420	0.3725	0.3042	0.2719	0.2201
53	0.0664	0.1585	0.3557	0.4187	0.3653	0.3258	0.3875	0.3952	0.3179	0.3295	0.3053	0.2412
54	0.0452	0.1762	0.3143	0.4164	0.3395	0.3127	0.4324	0.3208	0.3029	0.3263	0.3088	0.2297
55	0.0230	0.1508	0.3754	0.3610	0.3100	0.3729	0.3481	0.3436	0.3065	0.3079	0.3084	0.2300
56	0.0283	0.1146	0.3806	0.3481	0.3312	0.2503	0.4085	0.3293	0.3008	0.3054	0.2998	0.2163
57	0.0900	0.1080	0.3409	0.4287	0.2600	0.4010	0.3702	0.3977	0.4270	0.3256	0.2987	0.1707
58	0.0961	0.0905	0.3277	0.4139	0.2416	0.3790	0.3538	0.3776	0.4252	0.3222	0.2830	0.1523
59	0.0398	0.1024	0.3029	0.4743	0.3204	0.2903	0.3005	0.4575	0.3330	0.3894	0.2668	0.1587
60	0.0510	0.0465	0.2934	0.4674	0.2849	0.2721	0.3012	0.4498	0.3349	0.3276	0.3095	0.1313
61	0.0368	0.1097	0.3436	0.3658	0.1714	0.3584	0.4050	0.4060	0.4079	0.3220	0.2403	0.1573
62	0.0299	0.0997	0.3266	0.3630	0.1824	0.2693	0.4063	0.4266	0.3858	0.3211	0.2816	0.0960
63	0.0507	0.0317	0.3444	0.3796	0.1879	0.3389	0.3296	0.3893	0.4128	0.3216	0.2142	0.1403
64	0.0230	0.0415	0.2962	0.3960	0.0984	0.3361	0.3513	0.4040	0.3890	0.3080	0.2118	0.1212

Figure E.2: Vector Quantizer

V.No	ωε1	ωε2	ωε3	ωε4	ωε5	ωε6	ωε7	ωε8	ωε9	ωε10	ωε11	ωε12
65	0.1749	0.2711	0.3509	0.2832	0.3809	0.3271	0.3205	0.3073	0.2529	0.3150	0.3533	0.2636
66	0.1463	0.2639	0.3054	0.3037	0.3095	0.3583	0.3330	0.2943	0.2422	0.2925	0.3793	0.2209
67	0.1452	0.2395	0.3484	0.2310	0.3680	0.3805	0.2996	0.2515	0.2604	0.2173	0.4151	0.2469
68	0.1446	0.2253	0.3308	0.2106	0.3585	0.3193	0.3279	0.2472	0.2701	0.1931	0.4232	0.2162
69	0.1141	0.2203	0.3274	0.2827	0.3867	0.2804	0.2711	0.4062	0.2618	0.3015	0.3073	0.2378
70	0.1284	0.1839	0.3107	0.2793	0.3207	0.3166	0.2650	0.3504	0.2750	0.2645	0.3208	0.2213
71	0.0785	0.1926	0.3550	0.2562	0.3424	0.2281	0.3491	0.3224	0.2160	0.2681	0.3575	0.2406
72	0.0936	0.1792	0.3323	0.2452	0.3206	0.2288	0.2538	0.3906	0.2040	0.2835	0.3282	0.2296
73	0.1612	0.2152	0.2367	0.2929	0.2657	0.2934	0.3417	0.3551	0.2722	0.3097	0.2872	0.2190
74	0.1428	0.1904	0.2305	0.3084	0.2296	0.3213	0.3425	0.2786	0.2678	0.2826	0.3235	0.1741
75	0.1459	0.2080	0.2670	0.2201	0.2468	0.2830	0.3040	0.3371	0.2760	0.2751	0.2709	0.2309
76	0.1073	0.2191	0.2455	0.2228	0.2437	0.2830	0.2606	0.3413	0.2753	0.2623	0.2633	0.2200
77	0.0580	0.2067	0.2726	0.2860	0.3241	0.3120	0.2887	0.2887	0.2200	0.2312	0.3531	0.2297
78	0.0650	0.1860	0.2720	0.2659	0.3689	0.2971	0.3000	0.1941	0.2254	0.2209	0.3441	0.2080
79	0.0736	0.1601	0.2781	0.2435	0.2546	0.3476	0.2615	0.2499	0.2210	0.2806	0.2970	0.1811
80	0.0509	0.1651	0.2649	0.2232	0.3600	0.2765	0.2165	0.2479	0.2193	0.2632	0.2790	0.1876
81	0.1026	0.1393	0.3058	0.3518	0.2160	0.2063	0.4051	0.3413	0.3184	0.3214	0.2443	0.1887
82	0.0879	0.1204	0.3077	0.3386	0.1775	0.2152	0.3360	0.3952	0.3298	0.2880	0.2247	0.1963
83	0.0683	0.1439	0.2489	0.3856	0.1622	0.3211	0.2859	0.3545	0.3320	0.2755	0.2710	0.1321
84	0.0589	0.1245	0.2663	0.3522	0.0938	0.3519	0.2834	0.3203	0.3330	0.2519	0.2632	0.1284
85	0.0895	0.1063	0.3173	0.2953	0.2236	0.2629	0.3119	0.3142	0.2633	0.2874	0.2804	0.1573
86	0.0811	0.0950	0.2945	0.2956	0.2340	0.2263	0.3013	0.3158	0.2715	0.2695	0.2595	0.1666
87	0.0195	0.1464	0.2789	0.2988	0.1286	0.3637	0.2504	0.2798	0.2830	0.2414	0.2444	0.1804
88	0.0196	0.1238	0.2735	0.2830	0.1989	0.2352	0.2722	0.3143	0.2378	0.2513	0.2786	0.1260
89	0.0890	0.0805	0.2413	0.3974	0.2884	0.2430	0.2943	0.2868	0.2873	0.3450	0.2782	0.0709
90	0.0772	0.0730	0.2371	0.3726	0.2628	0.2159	0.2862	0.2855	0.2610	0.3457	0.2354	0.0724
91	0.0387	0.1013	0.2346	0.3492	0.2734	0.2600	0.3094	0.1956	0.3734	0.2781	0.1967	0.0767
92	0.0309	0.0731	0.2438	0.3207	0.2314	0.2435	0.2699	0.2375	0.3577	0.2770	0.1965	0.0511
93	0.0729	0.0688	0.1869	0.4045	0.2764	0.2643	0.1903	0.2992	0.3232	0.2467	0.1833	0.1736
94	0.0627	0.0518	0.1631	0.3979	0.2987	0.2359	0.1648	0.2696	0.2993	0.2306	0.1872	0.1528
95	0.0375	0.0300	0.2122	0.3593	0.3065	0.2723	0.2236	0.1564	0.2809	0.2617	0.1869	0.0886
96	0.0356	0.0299	0.1672	0.3420	0.2594	0.2033	0.2449	0.1964	0.2892	0.2504	0.1342	0.0905

Figure E.3: Vector Quantizer

V.No	ωε1	ωε2	ωε3	ωε4	ωε5	ωε6	ωε7	ωε8	ωε9	ωε10	ωε11	ωε12
97	0.1541	0.1699	0.2473	0.2158	0.1365	0.1826	0.3682	0.3442	0.1829	0.1881	0.3693	0.2262
98	0.1531	0.1597	0.2193	0.2271	0.1707	0.1805	0.3079	0.3387	0.1866	0.1900	0.3109	0.2357
99	0.1336	0.1388	0.1932	0.2808	0.1370	0.2366	0.1883	0.4114	0.1953	0.2209	0.3073	0.1739
100	0.1195	0.1323	0.1992	0.2433	0.0918	0.2201	0.2417	0.3473	0.1877	0.2012	0.3096	0.1478
101	0.1150	0.1725	0.2521	0.1635	0.1464	0.2839	0.2421	0.2857	0.2036	0.1460	0.2991	0.2212
102	0.0973	0.1476	0.2476	0.1662	0.1720	0.2296	0.2492	0.2784	0.1806	0.1452	0.2897	0.2141
103	0.1375	0.1443	0.1959	0.1708	0.1386	0.1914	0.3234	0.2651	0.1351	0.2148	0.2568	0.1838
104	0.1002	0.1382	0.2016	0.1516	0.0952	0.2140	0.2872	0.2237	0.1125	0.1953	0.2524	0.1789
105	0.0917	0.0943	0.2440	0.2790	0.2396	0.1878	0.2158	0.3262	0.2418	0.2341	0.2086	0.1896
106	0.0863	0.0856	0.2358	0.2621	0.2231	0.1585	0.1844	0.3239	0.2433	0.2089	0.2238	0.1505
107	0.0436	0.0960	0.2594	0.2355	0.2167	0.1373	0.2479	0.2508	0.2602	0.1772	0.1851	0.1717
108	0.0468	0.0695	0.2585	0.2110	0.1936	0.1618	0.1709	0.2669	0.2701	0.1812	0.1377	0.1637
109	0.0733	0.0839	0.1720	0.3113	0.1243	0.1091	0.3404	0.3135	0.2457	0.1927	0.2771	0.0657
110	0.0537	0.0693	0.1551	0.3084	0.0452	0.1696	0.2673	0.3113	0.2270	0.1830	0.2683	0.0535
111	0.0350	0.0455	0.2280	0.2369	0.0693	0.2362	0.2036	0.2527	0.2127	0.2272	0.1994	0.0564
112	0.0246	0.0379	0.2021	0.2296	0.0874	0.1887	0.1546	0.2963	0.2200	0.2133	0.1870	0.0163
113	0.1019	0.1210	0.1612	0.2564	0.1995	0.2587	0.1995	0.1806	0.2091	0.1683	0.1993	0.2019
114	0.0852	0.1168	0.1550	0.2352	0.2745	0.1957	0.1614	0.1432	0.2091	0.1303	0.2020	0.1853
115	0.1103	0.1255	0.1592	0.1680	0.1548	0.2155	0.2178	0.1370	0.1978	0.1824	0.1164	0.2228
116	0.0909	0.1118	0.1346	0.1923	0.1958	0.1948	0.1405	0.1483	0.2004	0.1478	0.1077	0.1984
117	0.0695	0.0968	0.1472	0.2539	0.1433	0.0875	0.2476	0.2854	0.1776	0.1677	0.2601	0.0875
118	0.0651	0.0800	0.1534	0.2258	0.0891	0.0281	0.3254	0.2225	0.1681	0.1236	0.2671	0.0677
119	0.0581	0.0625	0.1043	0.2879	0.0962	0.0970	0.1843	0.2554	0.2296	0.1691	0.1316	0.1172
120	0.0391	0.0356	0.1118	0.2602	0.0648	0.0228	0.1923	0.2486	0.1956	0.1446	0.1701	0.0629
121	0.0962	0.1005	0.1800	0.1295	0.1937	0.1024	0.1461	0.2290	0.1376	0.0781	0.2777	0.1860
122	0.0820	0.0795	0.1797	0.1209	0.1378	0.1224	0.1574	0.1805	0.1495	0.0682	0.1968	0.2124
123	0.0765	0.0755	0.1290	0.1785	0.2129	0.1435	0.1453	0.0915	0.0655	0.1361	0.2209	0.1543
124	0.0711	0.0597	0.1491	0.1247	0.1978	0.1059	0.0812	0.0729	0.0225	0.1064	0.2330	0.1403
125	0.0623	0.0630	0.0814	0.2217	0.0842	0.1345	0.0976	0.2424	0.1995	0.0833	0.1456	0.1359
126	0.0388	0.0330	0.0811	0.2094	0.1129	0.0722	0.0298	0.2664	0.1792	0.0877	0.1597	0.0718
127	0.0655	0.0579	0.1098	0.1075	0.0859	0.0552	0.2249	0.0878	0.1957	0.1078	0.0425	0.1350
128	0.0538	0.0299	0.0752	0.1052	0.0384	-0.0135	0.1188	0.1736	0.1158	0.0927	0.0829	0.0876

Figure E.4: Vector Quantizer

Bibliography

[1] T. S. Rappaport. *Wireless Communications.* Prentice Hall Publications, 1996.

[2] T. W. Parsons. *Voice and Speech Processing.* Prentice Hall Publications, 1987.

[3] J. G. Proakis, J. R. Deller and J. H. L. Hansen. *Discrete Time Processing of Speech Signals.* Macmillan Publishing, 1993.

[4] R.V.Cox and P.Kroon. "Low bit-rate speech coders for multimedia communication". *IEEE Communications Magazine*, 34(12):34–41, Dec. 1996.

[5] Luis Diez del Rio. "Secure speech and data communication over the public switching telephone network". In *Proceedings of ICASSP*, pages 425–428, 1994.

[6] J. P. Woodard and L. Hanzo. "Improvements to the analysis-by-synthesis loop in CELP". In *Conference on Radio Receivers and Associated Systems*, pages 114–118, 1995.

[7] P. Combescure, A. Kataoka, J. P. Adoul and P. Kroon. "Itu-t 8-kbit/s standard speech codec for Personal Communication Services". In *Int. Conference on Universal Personal Communication*, pages 818–822, 1995.

[8] J. Makhoul. "Linear prediction : A tutorial review". *IEEE Communications Magazine*, 63(4):561–580, Apr. 1975.

[9] F. Itakura and S. Saito. "A statistical method for estimation of speech spectral density and formant frequencies". *Trans. on Electronics and Communications in Japan*, 53-A(1), Jan. 1970.

[10] B. S. Atal and M. R. Schroeder. "Adaptive predictive coding of speech signals". *The Bell System Technical Journal*, pages 1973–1986, Oct. 1970.

[11] B. Atal and S. Hanauer. "Speech analysis and synthesis by linear prediction of the speech waves". *Journal of Acoustic Society of America*, 50(6):637–645, Oct. 1971.

[12] J. Markel and A. Gray. *Linear Prediction of Speech*. Springer-Verlag Publications, 1976.

[13] A. Oppenheim. "A speech analysis-synthesis system based on homomorphic filtering". *Journal of Acoustic Society of America*, 45(2), Feb. 1969.

[14] L. R. Rabiner and R. W. Schafer. *Digital Processing of Speech Signals*. Prentice Hall Publications, 1978.

[15] B. Atal and J. Remde. "A new model for LPC excitation for producing natural sounding speech at low bit rates". In *Proceedings of ICASSP*, pages 614–617, 1982.

[16] A. Buzo, Y. Linde and R. M. Gray. "An algorithm for vector quantizer design". *IEEE transactions on Communications*, 28(1):84–95, Jan. 1980.

[17] K. K. Paliwal and B. S. Atal. "Efficient vector quantization of LPC parameters at 24 bits/frame". In *Proceedings of ICASSP*, pages 661–664, 1991.

[18] W. R. Gardner and B. D. Rao. "Theoretical analysis of the high-rate vector quantization of LPC parameters". *IEEE transactions on Speech and Audio Processing*, 3(5):367–381, Sep. 1995.

[19] S. Bruhn. "Efficient interblock noiseless coding of speech LPC parameters". In *Proceedings of ICASSP*, pages I/501–503, 1994.

[20] H. Wakita. "Linear prediction voice synthesizers: Line spectrum pairs is the newest of several techniques". *Speech Technology*, pages 17–22, Fall. 1981.

[21] F. K. Soong and B. Juang. "Line spectrum pair and speech data compression". In *Proceedings of ICASSP*, pages 1.10.1–1.10.4, 1984.

[22] J. R. Crosmer. *Very low bit rate speech coding using the line spectrum pair transformation of the LPC coefficients*. PhD thesis, Georgia Institute of Technology, 1985.

[23] R. Schroeder and B. S. Atal. "Code Excited Linear Prediction (CELP): High quality speech at low bit rates". In *Proceedings of ICASSP*, pages 1647–1650, 1985.

[24] N. Sugamura and F. Itakura. "Speech analysis and synthesis methods developed at ECL in NTT". *Speech Communications*, (5):199–215, 1986.

[25] J. P. Campbell. *"The DOD 4.8 Kbps Standard"*. US Government, Department of Defence, 1988.

[26] Peter Noll. "Mpeg digital audio coding". *IEEE Signal Processing Magazine*, 14(5):59-81, Sep. 1997.

[27] R. V. Cox. *"Draft Recommendation AV. 25Y - dual rate speech coder for multimedia telecommunications (G.723.1)"*. ITU-T Standardization sector, 1995.

[28] D. Massaloux and S Proust. "Spectral shaping in the proposed ITU-T 8 kb/s speech coding standard". In *IEEE Speech Coding Workshop*, pages 9–10, 1995.

[29] J. Ikedo, A. Kataoka and S. Hayashi. "Lsp and gain quantization for the proposed ITU-T 8 kb/s speech coding standard". In *IEEE Speech Coding Workshop*, pages 7–8, 1995.

[30] J. P. Adoul, R. Salami, C. Laflamme and D. Massaloux. "A toll quality 8kb/s speech codec for the Personal Communications System (PCS)". *IEEE transactions on Vehicular Technology*, 43(3):808–816, Aug. 1994.

[31] T. Moriya, A. Kataoka and S. Hayashi. "An 8-kb/s Conjugate Structure CELP (CS-CELP) speech coder". *IEEE transactions on Speech and Audio processing*, 4(6):401–411, Nov. 1996.

[32] T. Moriya. "Two-channel conjugate vector quantizer for noisy channel speech coding". *IEEE Journal on Areas in Communication*, 10(5):865–873, June. 1992.

[33] R. P. Ramachandran et al. "A two codebook formant for robust quantization of line spectral frequencies". *IEEE transactions on Speech and Audio Processing*, 3(3):157–167, May. 1995.

[34] C. Deyuan. "An 8 kb/s low complexity ACELP speech codec". In *Proceedings of ICSP*, pages 671-674, 1996.

[35] C. Laflamme, R. Salami and J. P. Adoul. "8 kbit/s ACELP coding of speech with 10 ms speech-frame: A candidate for CCITT standardization". In *Proceedings of ICASSP*, pages II /97-100, 1992.

[36] F. Itakura. "Maximum prediction residual principle applied to speech recognition". *IEEE transactions on Accoustics, Speech and Audio Processing*, 23(1):67–72, Feb. 1975.

[37] R. Steele. *Mobile Radio Communications*. Pentech Press, 1992.

[38] C. Laflamme, et al. "16 kbps wideband speech coding technique based on algebraic CELP". In *Proceedings of ICASSP*, pages 177–180, 1991.

[39] K. Soo-Nee, C. Chang-Qian and S. Pratab. "A modified generalised Lloyd algorithm for VQ codebook design". In *Proceedings of ICASSP*, pages 542–545, 1996.

[40] P. Kabal and R. P. Ramachandran. "The computation of line spectral frequencies using Chebyshev polynomials". *IEEE transactions on Accoustics, Speech and Speech Processing*, 34(6):1419–1425, Dec. 1986.

[41] M. Ansorge, S. Grassi, A. Dufaux and F. Pellandini. "Efficient algorithm to compute LSP parameters from 10th-order LPC Coefficients". In *Proceedings of ICASSP*, pages 1707–1710, 1997.

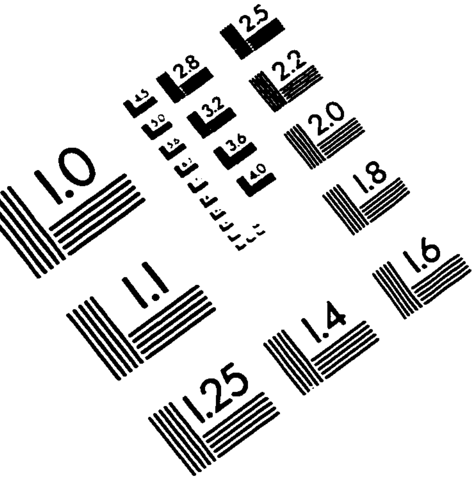
[42] Sang-Min Lee and Sangil Park. "New implementation of G.723.1 on vector processor for video conferencing system". In *IEEE International Conference on Consumer Electronics*, pages 318–319, 1997.

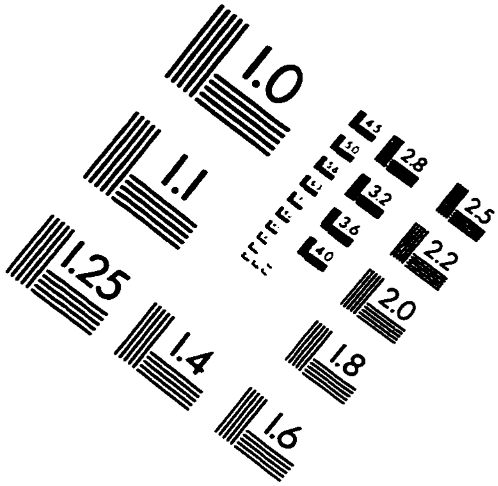
[43] Y. Cho, D. Chang and S. Ann. "Efficient quantization of LSF parameters using classified SVQ combined with conditional splitting". In *Proceedings of ICSP*, pages 671–674, 1996.

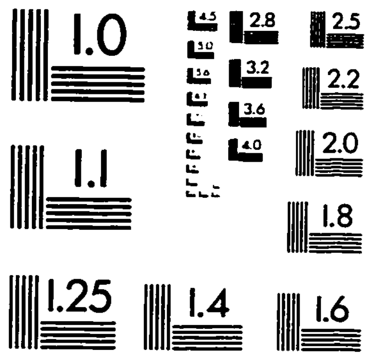
Vita

- Shahid Parvez
- Born in Bombay, India on September 25, 1973
- Received Bachelor's degree in Electronics and Communication Engineering from Osmania University, Hyderabad, India in July, 1995.
- Completed Master's degree requirements at King Fahd University of Petroleum and Minerals, Dhahran, Saudi Arabia in May, 1998.

IMAGE EVALUATION TEST TARGET (QA-3)

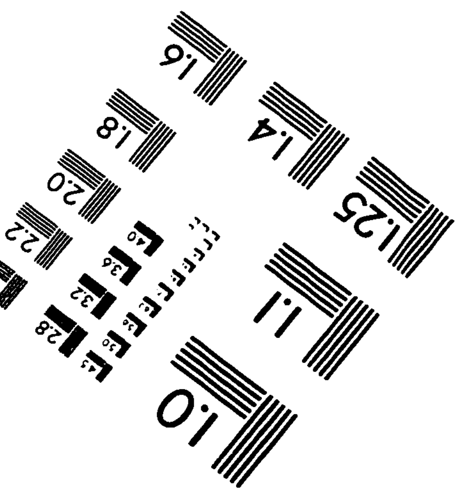


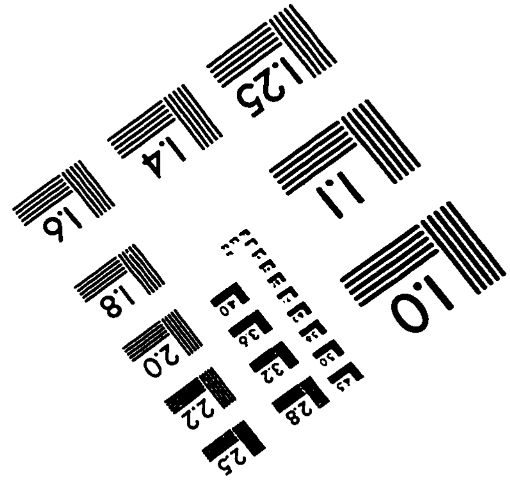




150mm

6"





APPLIED IMAGE . Inc
1653 East Main Street
Rochester, NY 14609 USA
Phone: 716/482-0300
Fax: 716/288-5989